# SAMA5D2 Series

## Ultra-Low-Power Arm® Cortex®-A5 Core-Based MPU, 500 MHz, Graphics Interface, Ethernet 10/100, CAN, USB, PCI 5.0 Pre-Certified

## Introduction

The SAMA5D2 series is a high-performance, ultra-low-power Arm Cortex-A5 CPU-based embedded microprocessor (MPU) running up to 500 MHz, with support for multiple memories such as DDR2, DDR3L, LPDDR2, LPDDR3, and QSPI and e.MMC Flash. The devices integrate powerful peripherals for connectivity and user interface applications, and offer advanced security functions (Arm TrustZone®, tamper detection, secure data storage, etc.), as well as high-performance crypto accelerators AES, SHA and TRNG.

Selected members of the SAMA5D2 series are qualified for extended industrial temperature range operation (-40°C to 105°C external temperature).

The SAMA5D2 series is delivered with a free Linux® distribution and bare metal C examples.

## Features

- Arm Cortex-A5 Core
  - Armv7-A architecture
  - Arm TrustZone
  - NEON™ Media Processing Engine
  - Up to 500 MHz
  - ETM/ETB 8 Kbytes
- Memory Architecture
  - Memory Management Unit (MMU)
  - 32-Kbyte L1 data cache, 32-Kbyte L1 instruction cache
  - 128-Kbyte L2 cache configurable to be used as an internal SRAM
  - One 128-Kbyte scrambled internal SRAM
  - One 160-Kbyte internal ROM
    - 64-Kbyte scrambled and maskable ROM embedding bootloader/Secure bootloader
    - 96-Kbyte unscrambled, unmaskable ROM for NAND Flash BCH ECC table
  - High-bandwidth scramblable 16-bit or 32-bit Double Data Rate (DDR) multiport dynamic RAM controller supporting up to 512 Mbytes 8-bank DDR2/DDR3 (DLL off only) / DDR3L (DLL off only) / LPDDR1/LPDDR2/LPDDR3, including “on-the-fly” encryption/decryption path
  - 8-bit SLC/MLC NAND controller, with up to 32-bit Error Correcting Code (PMECC)
- System Running up to 166 MHz in Typical Conditions

© 2018 Microchip Technology Inc.

  - Reset Controller (RSTC), Shutdown Controller (SHDWC), Periodic Interval Timer (PIT), independent Watchdog Timer (WDT) and secure Real-Time Clock (RTC) with clock calibration
  - One 600 to 1200 MHz PLL for the system and one 480 MHz PLL optimized for high-speed USB
  - Digital fractional PLL for audio (11.2896 MHz and 12.288 MHz)
  - Internal low-power 12 MHz RC and 32 kHz typical RC
  - Selectable 32.768 Hz low-power oscillator and 8 to 24 MHz oscillator
  - 51 DMA channels including two 16-channel 64-bit Central DMA Controllers
  - 64-bit Advanced Interrupt Controller (AIC)
  - 64-bit Secure Advanced Interrupt Controller (SAIC)
  - Three programmable external clock signals
- Low-Power Modes
  - Ultra-low-power mode with fast wake-up capability
  - Low-power Backup mode with 5-Kbyte SRAM and SleepWalking™ features
    - Wake up from up to nine wake-up pins, UART reception, analog comparison
    - Fast wake-up capability
    - Extended Backup mode with DDR in Self-Refresh mode
- Peripherals
  - LCD TFT controller (LCDC) up to 1024x768 or 1280x768 (still image). Four overlays, rotation, post-processing and alpha blending, 24-bit parallel RGB interface
  - ITU-R BT. 601/656/1120 Image Sensor Controller (ISC) supporting up to 5 Mpixel sensors with a parallel 12-bit interface for Raw Bayer, YCbCr, Monochrome and JPEG-compressed sensor interface
  - Two Synchronous Serial Controllers (SSC), two Inter-IC Sound Controllers (I2SC), and one Stereo Class D amplifier (CLASSD)
  - One Peripheral Touch Controller (PTC) with up to 8 X-lines and 8 Y-lines (64-channel capacitive touch)
  - One Pulse Density Modulation Interface Controller (PDMIC)
  - One USB device high-speed port (UDPHS) and one USB host high-speed port or two USB host high-speed ports (UHPHS)
  - One USB host high-speed port with a High-Speed Inter-Chip (HSIC) interface
  - One 10/100 Ethernet MAC (GMAC)
    - Energy efficiency support (IEEE® 802.3az standard)
    - Ethernet AVB support with IEEE802.1AS timestamping
    - IEEE802.1Qav credit-based traffic-shaping hardware support
    - IEEE1588 Precision Time Protocol (PTP)
  - Two high-speed memory card hosts:
    - SDMMC0: SD 3.0, eMMC 4.51, 8 bits
    - SDMMC1: SD 2.0, eMMC 4.41, 4 bits only
  - Two master/slave Serial Peripheral Interfaces (SPI)
  - Two Quad Serial Peripheral Interfaces (QSPI)
  - Five FLEXCOMs (USART, SPI and TWI)
  - Five UARTs
  - Two master CAN-FD (MCAN) controllers with SRAM-based mailboxes, and time- and event-triggered transmission

**WARNING** MCAN implements the non-ISO CAN FD frame format and therefore does not pass the CAN FD Conformance Test according to ISO 16845-1:2016.

  - One Rx only UART in backup area (RXLP)
  - One Analog Comparator Controller (ACC) in backup area
  - Two 2-wire interfaces (TWIHS) up to 400 Kbits/s supporting the $I^2C$ protocol and SMBUS
  - One full-featured 4-channel 16-bit Pulse Width Modulation (PWM) controller
  - Two 3-channel 32-bit Timer/Counters (TC), supporting basic PWM modes
  - One 12-channel, 12-bit, Analog-to-Digital Converter (ADC) with resistive touchscreen capability
- Safety
  - Zero-power Power-on Reset (POR) cells
  - Main crystal clock failure detector
  - Write-protected registers
  - Integrity Check Monitor (ICM) based on SHA256
  - Memory Management Unit (MMU)
  - Independent watchdog
- Security
  - 5 Kbytes of internal scrambled SRAM:
    - 1 Kbyte nonerasable on tamper detection
    - 4 Kbytes erasable on tamper detection
  - 256 bits of scrambled and erasable registers
  - Up to eight tamper pins for static or dynamic intrusion detections(1)
  - Environmental monitors on specific versions: temperature, voltage, frequency and active die shield(2)
  - Secure Bootloader(3)
  - On-the-fly AES encryption/decryption on DDR and QSPI memories (AESB)
  - RTC including timestamping on security intrusions
  - Programmable fuse box with 544 fuse bits (including JTAG protection and BMS)

  **Note:**
  1. For information specific to dynamic tamper protection (PIOBU), refer to the document *SAMA5D2External Tamper Protections* (document no. 44095).
  2. For environmental monitors, refer to the document *SAMA5D23 and SAMA5D28 Environmental Monitors* (document no. 44036), available under Non-Disclosure Agreement (NDA). Contact a Microchip Sales Representative for details.
  3. For secure boot strategies, refer to the document *SAMA5D2 Series Secure Boot Strategy* (document no. 44040), available under Non-Disclosure Agreement (NDA). Contact a Microchip Sales Representative for details.
- Hardware Cryptography
  - SHA (SHA1, SHA224, SHA256, SHA384, SHA512): compliant with FIPS PUB 180-2
  - AES: 256-, 192-, 128-bit key algorithms, compliant with FIPS PUB 197
  - TDES: two-key or three-key algorithms, compliant with FIPS PUB 46-3
  - True Random Number Generator (TRNG) compliant with NIST Special Publication 800-22 Test Suite and FIPS PUBs 140-2 and 140-3
- Up to 128 I/Os

© 2018 Microchip Technology Inc.

  - Fully programmable through set/clear registers
  - Multiplexing of up to eight peripheral functions per I/O line
  - Each I/O line can be assigned to a peripheral or used as a general-purpose I/O
  - The PIO controller features a synchronous output providing up to 32 bits of data output in one write operation
- Packages
  - 289-ball LFBGA, 14 x 14 mm body, 0.8 mm pitch
  - 256-ball TFBGA, 8 x 8 mm body, 0.4 mm pitch
  - 196-ball TFBGA, 11 x 11 mm body, 0.75 mm pitch

# 1. Description

The SAMA5D2 Series is a high-performance, power-efficient embedded MPU based on the Arm Cortex-A5 processor. It integrates the Arm NEON SIMD engine for accelerated multimedia and signal processing, a configurable 128-Kbyte L2 cache and a floating point unit (FPU) for high-precision computing. The device features an advanced user interface and connectivity peripherals. Advanced security is provided by powerful cryptographic accelerators, by the Arm TrustZone technology securing access to memories and sensitive peripherals, and by several hardware features that safeguard memory content, authenticate software, detect physical attacks and prevent information leakage during code execution.

The SAMA5D2 features an internal multilayer bus architecture associated with 2 x 16 DMA channels and dedicated DMAs for the communication and interface peripherals required to ensure uninterrupted data transfers with minimal processor overhead. The device supports DDR2, DDR3, DDR3L, LPDDR1, LPDDR2, LPDDR3, QSPI and e.MMC Flash, and SLC/MLC parallel NAND Flash memory up to 32-bit ECC.

The comprehensive peripheral set includes an LCD TFT controller with overlays for hardware-accelerated image composition, an image sensor controller, audio support through $I^2S$, SSC, a stereo Class D amplifier and a digital microphone. Connectivity peripherals include a 10/100 EMAC, USBs, CANs, FLEXCOMs, UARTs, SPIs and two QSPIs, SDIO/SD/e.MMCs, and TWIs/$I^2C$.

Protection of code and data is provided by automatic scrambling of memories and an Integrity Check Monitor (ICM) to detect any modification of the memory contents. The SAMA5D2 also supports execution of encrypted code (QSPI or one portion of the DDR) with an “on-the-fly” encryption-decryption process.

With its secure design architecture, cryptographic acceleration engines, and secure bootloader, the SAMA5D2 is the ideal solution for point-of-sale (POS), IoT and industrial applications requiring device authentication, anti-cloning, data protection and secure communication.

SAMA5D2 devices feature three software-selectable low-power modes: Idle, Ultra-Low-Power and Backup.

In Idle mode, the processor is stopped while all other functions can be kept running.

In Ultra-Low-Power mode 0, the processor is stopped while all other functions are clocked at 512 Hz and interrupts or peripherals can be configured to wake up the system based on events, including partial asynchronous wake-up (SleepWalking).

In Ultra-Low-Power mode 1, all clocks and functions are stopped but some peripherals can be configured to wake up the system based on events, including partial asynchronous wake-up (SleepWalking).

In Backup mode, RTC and wake-up logic are active. The Backup mode can be extended to feature DDR in Self-refresh mode.

SAMA5D2 devices also include an Event System that allows peripherals to receive, react to and send events in Active and Idle modes without processor intervention.

© 2018 Microchip Technology Inc.

# 2. Configuration Summary

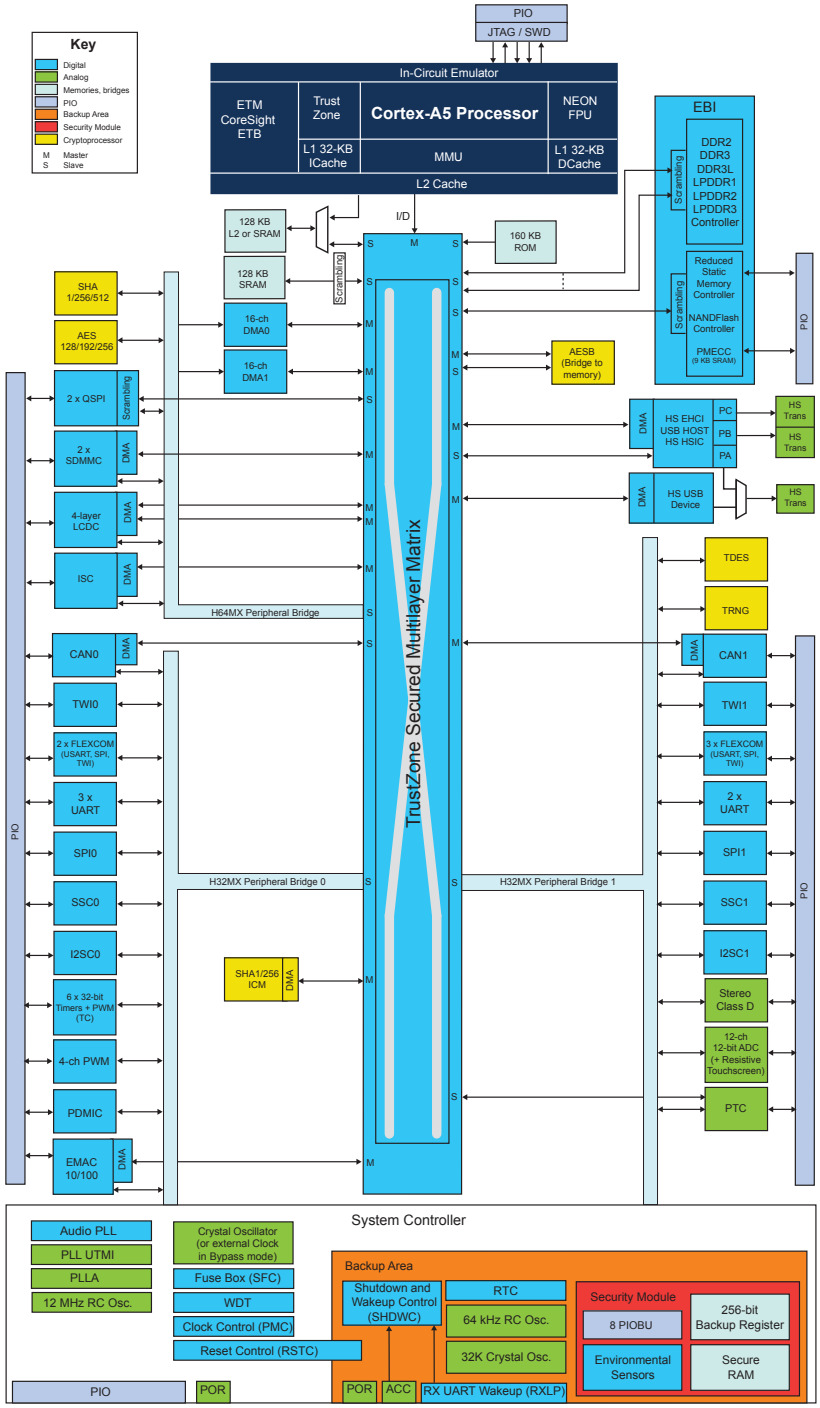
**Table 2-1. SAMA5D2 Configuration Summary**

| Feature | SAMA5D21 | SAMA5D22 | SAMA5D23 | SAMA5D24 | SAMA5D26 | SAMA5D27 | SAMA5D28 |
|---|---|---|---|---|---|---|---|
| Package | TFBGA196 | | | TFBGA256 | LFBGA289 | | |
| PIOs | 72 | | | 105 | 128 | | |
| DDR Bus | 16-bit | | | 16/32-bit | | | |
| SMC | Up to 16-bit | | | | | | |
| SRAM | 128 Kbytes | | | | | | |
| QSPI | 2 | | | | | | |
| LCD | 24-bit RGB | | | | | | |
| Camera Interface (ISC) | 1 | | | | | | |
| EMAC | 1 | | | | | | |
| PTC | – | 4 X-lines x 8 Y-lines | | 8 X-lines x 8 Y-lines | – | 8 X-lines x 8 Y-lines | |
| CAN | – | 1 | | – | | 2 | |
| USB | 2 (2 Hosts or 1 Host/1 Device) | | | 3 (2 Hosts/ 1 HSIC, or 1 Host/ 1 Device/ 1 HSIC) | 2 (2 Hosts or 1 Host/ 1 Device) | 3 (2 Hosts/1 HSIC or 1 Host/1 Device/1 HSIC) | |
| UART/SPI/$I^2C$ | 9 / 6 / 6 | | | 10 / 7 / 7 | | | |
| SDIO/SD/MMC | 1 | | | 2 | | | |
| $I^2S$/SSC/ Class D/PDM | 2 / 2 / 1 / 1 | | | | | | |
| ADC Inputs | 5 | | | 12 | | | |
| Timers | 5 | | | 6 | | | |
| PWM | 4 (PWM) + 5 (TC) | | | 4 (PWM) + 6 (TC) | | | |
| Tamper Pins | 6 | | | 2 | 8 | | |
| AESB | – | Yes | | | – | Yes | |
| Environmental Monitors, Die Shield | – | | Yes | – | | | Yes |

For information on device pin compatibility, see the section "Pinouts".

# 3. Block Diagram

**Figure 3-1. SAMA5D2 Series Block Diagram**




Refer to the section DMA Controller (XDMAC) for peripheral connections to DMA.

# 4. Signal Description

**Table 4-1. Signal Description List**

| Signal Name | Function | Type | Comments | Active Level |
|---|---|---|---|---|
| **Clocks, Oscillators and PLLs** | | | | |
| XIN | Main Oscillator Input | Input | – | – |
| XOUT | Main Oscillator Output | Output | – | – |
| XIN32 | Slow Clock Oscillator Input | Input | – | – |
| XOUT32 | Slow Clock Oscillator Output | Output | – | – |
| CLK_AUDIO | Audio Clock | Output | – | – |
| VBG | Bias Voltage Reference for USB | Analog | – | – |
| PCK 0–2 | Programmable Clock Output | Output | Reset State:<br>- PIO Input<br>- Internal Pull-up enabled<br>- Schmitt Trigger enabled | – |
| **Shutdown, Wake-up Logic** | | | | |
| SHDN | Shutdown Control | Output | – | – |
| PIOBU 0–7 | Tamper or Wake-up Inputs | Input | – | – |
| WKUP | Wake-up Input | Input | – | – |
| **ICE and JTAG** | | | | |
| TCK/SWCLK | Test Clock/Serial Wire Clock | Input | – | – |
| TDI | Test Data In | Input | – | – |
| TDO | Test Data Out | Output | – | – |
| TMS/SWDIO | Test Mode Select/Serial Wire Input/ Output | I/O | – | – |
| JTAGSEL | JTAG Selection | Input | – | – |
| **Reset/Test** | | | | |
| NRST | Microprocessor Reset | Input | – | Low |
| TST | Test Mode Select | Input | – | – |
| NTRST | Test Reset Signal | Input | – | – |
| **Advanced Interrupt Controller - AIC** | | | | |
| IRQ | External Interrupt Input | Input | – | – |
| **Secured Advanced Interrupt Controller - SAIC** | | | | |

.....continued

| Signal Name | Function | Type | Comments | Active Level |
|---|---|---|---|---|
| FIQ | Fast Interrupt Input | Input | – | – |
| **PIO Controller** | | | | |
| PA0–PA31 | Parallel IO Controller | I/O | – | – |
| PB0–PB31 | Parallel IO Controller | I/O | – | – |
| PC0–PC31 | Parallel IO Controller | I/O | – | – |
| PD0–PD31 | Parallel IO Controller | I/O | – | – |
| **External Bus Interface - EBI** | | | | |
| D[15:0] | Data Bus | I/O | – | – |
| A[25:0] | Address Bus | Output | – | – |
| NWAIT | External Wait Signal | Input | – | Low |
| **Static Memory Controller - HSMC** | | | | |
| NCS0–NCS3 | Chip Select Lines | Output | – | Low |
| NWR0–NWR1 | Write Signal | Output | – | Low |
| NRD | Read Signal | Output | – | Low |
| NWE | Write Enable | Output | – | Low |
| NBS0–NBS1 | Byte Mask Signal | Output | – | Low |
| NANDOE | NAND Flash Output Enable | Output | – | Low |
| NANDWE | NAND Flash Write Enable | Output | – | Low |
| **DDR2/DDR3/LPDDR1/LPDDR2/LPDDR3 Controller** | | | | |
| DDR_CK, DDR_CLKN | DDR Differential Clock | Output | – | – |
| DDR_CKE | DDR Clock Enable | Output | When Backup Self-refresh mode is used, should be tied to GND using 100 KΩ pull-down | High |
| DDR_CS | DDR Controller Chip Select | Output | – | Low |
| DDR_BA[2:0] | Bank Select | Output | – | Low |
| DDR_WE | DDR Write Enable | Output | – | Low |
| DDR_RAS, DDR_CAS | Row and Column Signal | Output | – | Low |
| DDR_A[13:0] | DDR Address Bus | Output | – | – |
| DDR_D[31:0] | DDR Data Bus | I/O/-PD | – | – |

.....continued

| Signal Name | Function | Type | Comments | Active Level |
|---|---|---|---|---|
| DDR_DQS[3:0], DDR_DQSN[3:0] | Differential Data Strobe | I/O- PD | – | – |
| DDR_DQM[3:0] | Write Data Mask | Output | – | – |
| DDR_CAL | DDR/LPDDR Calibration | Input | – | – |
| DDR_VREF | DDR/LPDDR Reference | Input | – | – |
| DDR_RESETN | DDR3 Active Low Asynchronous Reset | Output | When Backup Self-refresh mode is used, should be tied to VDDIODDR using 100 KΩ pull-up | – |
| **Secure Data Memory Card - SDMMCx [1:0]** | | | | |
| SDMMCx_CD | SDcard / e.MMC Card Detect | Input | – | – |
| SDMMCx_CMD | SDcard / e.MMC Command line | I/O | – | – |
| SDMMCx_WP | SDcard Connector Write Protect Signal | Input | – | – |
| SDMMCx_RSTN | e.MMC Reset Signal | Output | – | – |
| SDMMCx_1V8SEL | SDcard Signal Voltage Selection | Output | – | – |
| SDMMCx_CK | SDcard / e.MMC Clock Signal | Output | – | – |
| SDMMCx_DAT[7:0] | SDcard / e.MMC Data Lines | I/O | – | – |
| **Flexible Serial Communication Controller - FLEXCOMx [4:0]** | | | | |
| FLEXCOMx_IO0 | FLEXCOMx Transmit Data | I/O | – | – |
| FLEXCOMx_IO1 | FLEXCOMx Receive Data | I/O | – | – |
| FLEXCOMx_IO2 | FLEXCOMx Serial Clock | I/O | – | – |
| FLEXCOMx_IO3 | FLEXCOMx Clear To Send / Peripheral Chip Select | I/O | – | – |
| FLEXCOMx_IO4 | FLEXCOMx Request To Send / Peripheral Chip Select | Output | – | – |
| **Universal Asynchronous Receiver Transmitter - UARTx [4..0]** | | | | |
| UTXDx | UARTx Transmit Data | Output | – | – |
| URXDx | UARTx Receive Data | Input | – | – |
| **Inter-IC Sound Controller - I2SCx [1..0]** | | | | |
| I2SCx_MCK | Master Clock | Output | – | – |
| I2SCx_CK | Serial Clock | I/O | – | – |
| I2SCx_WS | $I^2S$ Word Select | I/O | – | – |
| I2SCx_DI0 | Serial Data Input | Input | – | – |

.....continued

| Signal Name | Function | Type | Comments | Active Level |
|---|---|---|---|---|
| I2SCx_DO0 | Serial Data Output | Output | – | – |
| **Synchronous Serial Controller - SSCx [1..0]** | | | | |
| TDx | SSC Transmit Data | Output | – | – |
| RDx | SSC Receive Data | Input | – | – |
| TKx | SSC Transmit Clock | I/O | – | – |
| RKx | SSC Receive Clock | I/O | – | – |
| TFx | SSC Transmit Frame Sync | I/O | – | – |
| RFx | SSC Receive Frame Sync | I/O | – | – |
| **Timer/Counter - TCx [1..0]** | | | | |
| TCLK[5..0] | TC Channel y External Clock Input | Input | – | – |
| TIOA[5..0] | TC Channel y I/O Line A | I/O | – | – |
| TIOB[5..0] | TC Channel y I/O Line B | I/O | – | – |
| **Quad IO SPI - QSPIx [1..0]** | | | | |
| QSPIx_SCK | QSPI Serial Clock | Output | – | – |
| QSPIx_CS | QSPI Chip Select | Output | – | – |
| QSPIx_IO[0..3] | QSPI I/O<br>QIO0 is QMOSI Master Out - Slave In<br>QIO1 is QMISO Master In - Slave Out | I/O | – | – |
| **Serial Peripheral Interface - SPIx [1..0]** | | | | |
| SPIx_MISO | Master In Slave Out | I/O | – | – |
| SPIx_MOSI | Master Out Slave In | I/O | – | – |
| SPIx_SPCK | SPI Serial Clock | I/O | – | – |
| SPIx_NPCS0 | SPI Peripheral Chip Select 0 | I/O | – | Low |
| SPIx_NPCS[3..1] | SPI Peripheral Chip Select | Output | – | Low |
| **Two-wire Interface - TWIx [1..0]** | | | | |
| TWDx | Two-wire Serial Data | I/O | – | – |
| TWCKx | Two-wire Serial Clock | I/O | – | – |
| **Pulse Width Modulation Controller - PWM** | | | | |
| PWMH0–3 | PWM Waveform Output High | Output | – | – |
| PWML0–3 | PWM Waveform Output Low | Output | – | – |
| PWMFI0–1 | PWM Fault Inputs | Input | – | – |

.....continued

| Signal Name | Function | Type | Comments | Active Level |
|---|---|---|---|---|
| PWMEXTRG1–2 | PWM External Trigger | Input | – | – |
| **USB Host High-Speed Port - UHPHS** | | | | |
| HHSDPA | USB Host Port A High-Speed Data + | Analog | – | – |
| HHSDMA | USB Host Port A High-Speed Data - | Analog | – | – |
| HHSDPB | USB Host Port B High-Speed Data + | Analog | – | – |
| HHSDMB | USB Host Port B High-Speed Data - | Analog | – | – |
| **USB Device High-Speed Port - UDPHS** | | | | |
| DHSDP | USB Device High-Speed Data + | Analog | – | – |
| DHSDM | USB Device High-Speed Data - | Analog | – | – |
| **USB High-Speed Inter-Chip Port - HSIC** | | | | |
| HHSTROBE | USB High-Speed Inter-Chip Strobe | I/O | – | – |
| HHDATA | USB High-Speed Inter-Chip Data | I/O | – | – |
| **Ethernet 10/100 - GMAC** | | | | |
| GREFCK | Reference Clock | Input | – | – |
| GTXCK | Transmit Clock | Input | – | – |
| GRXCK | Receive Clock | Input | – | – |
| GTXEN | Transmit Enable | Output | – | – |
| GTX0–GTX3 | Transmit Data | Output | – | – |
| GTXER | Transmit Coding Error | Output | – | – |
| GRXDV | Receive Data Valid | Input | – | – |
| GRX0–GRX3 | Receive Data | Input | – | – |
| GRXER | Receive Error | Input | – | – |
| GCRS | Carrier Sense | Input | – | – |
| GCOL | Collision Detected | Input | – | – |
| GMDC | Management Data Clock | Output | – | – |
| GMDIO | Management Data Input/Output | I/O | – | – |
| GTSUCOMP | TSU timer comparison valid | Output | – | – |
| **LCD Controller - LCDC** | | | | |
| LCDDAT[23:0] | LCD Data Bus | Output | – | – |
| LCDVSYNC | LCD Vertical Synchronization | Output | – | – |

.....continued

| Signal Name | Function | Type | Comments | Active Level |
|---|---|---|---|---|
| LCDHSYNC | LCD Horizontal Synchronization | Output | – | – |
| LCDPCK | LCD Pixel Clock | Output | – | – |
| LCDDEN | LCD Data Enable | Output | – | – |
| LCDPWM | LCDPWM for Contrast Control | Output | – | – |
| LCDDISP | LCD Display ON/OFF | Output | – | – |
| **Touchscreen Analog-to-Digital Converter - ADC** | | | | |
| AD0–11 | 12 Analog Inputs | Analog | – | – |
| ADTRG | ADC Trigger | Input | – | – |
| ADVREF | ADC Reference | Analog | – | – |
| **Secure Box Module - SBM** | | | | |
| PIOBU0–7 | Tamper I/Os | I/O | – | – |
| **Image Sensor Controller - ISC** | | | | |
| ISC_D0–ISC_D11 | Image Sensor Data | Input | – | – |
| ISC_HSYNC | Image Sensor Horizontal Synchro | Input | – | – |
| ISC_VSYNC | Image Sensor Vertical Synchro | Input | – | – |
| ISC_PCK | Image Sensor Pixel clock | Input | – | – |
| ISC_MCK | Image Sensor Main clock | Output | – | – |
| ISC_FIELD | Field identification signal | Input | – | – |
| **Audio Class Amplifier - CLASSD** | | | | |
| CLASSD_L0 | CLASSD Left Output L0 | Output | – | – |
| CLASSD_L1 | CLASSD Left Output L1 | Output | – | – |
| CLASSD_L2 | CLASSD Left Output L2 | Output | – | – |
| CLASSD_L3 | CLASSD Left Output L3 | Output | – | – |
| CLASSD_R0 | CLASSD Right Output R0 | Output | – | – |
| CLASSD_R1 | CLASSD Right Output R1 | Output | – | – |
| CLASSD_R2 | CLASSD Right Output R2 | Output | – | – |
| CLASSD_R3 | CLASSD Right Output R3 | Output | – | – |
| **Control Area Network - CAN** | | | | |
| CANRXx | CAN Receive | Input | – | – |
| CANTXx | CAN Transmit | Output | – | – |

.....continued

| Signal Name | Function | Type | Comments | Active Level |
|---|---|---|---|---|
| **Peripheral Touch Controller - PTC** | | | | |
| PTC_X[7..0] | X-lines | Output | – | – |
| PTC_Y[7..0] | Y-lines | Input | – | – |
| **Pulse Density Modulation Interface Controller - PDMIC** | | | | |
| PDMIC_DAT | PDM Data | Input | – | – |
| PDMIC_CLK | PDM Clock | Output | – | – |

# 5. Safety and Security Features

## 5.1 Design for Safety and IEC60730 Class B Certification

### 5.1.1 Background Information

The IEC 60730 standard encompasses all aspects of appliance design. Annex H of the standard covers the aspects most relevant to microcontrollers. It details the tests and diagnostics which are intended to ensure safe operation of embedded control hardware and software. IEC 60730 defines three classifications for electronic control functions:

- Class A - Control functions which are not intended to be relied upon for safety of the equipment
- Class B - Control functions intended to prevent unsafe operation of the controlled equipment
- Class C - Control functions intended to prevent special hazards such as explosions

Specific design techniques have been used in the SAMA5D2 to ease compliance with the IEC 60730 Class B Certification and to resolve general-purpose safety concerns. This allows reduced software development and code size as well as savings on external hardware circuitry, since built-in self-tests are already embedded in the MPU. The table “Safety and IEC 60730 Features List” below gives the list of peripherals which incorporate these techniques, and details whether these features are applicable for the IEC 60730 Class B Certification or for general-purpose safety considerations.

## 5.2 Design for Security

The SAMA5D2 embeds peripherals with security features to prevent counterfeiting, to secure external communication, and to authenticate the system.

The table “Security Features” provides the list of peripherals and an overview of their security function. For more information, see the sections on each peripheral.

## 5.3 Safety and IEC 60730 Features

**Table 5-1. Safety and IEC 60730 Features List**

| Peripheral | Component | Fault/Error/Feature | Requirements for Class B IEC 60730(1) | General Safety |
|---|---|---|---|---|
| PMC | Clock | CPU clock monitoring<br>- Overclocking detection | – | X |
| | | 32.768 kHz crystal oscillator frequency monitoring<br>- Abnormal frequency deviation | X | X |
| | | Main crystal oscillator<br>- Crystal failure detection | X | X |

.....continued

| Peripheral | Component | Fault/Error/Feature | Requirements for Class B IEC 60730(1) | General Safety |
|---|---|---|---|---|
| PIOC | I/O Periphery | Programmable configuration lock (active until next $V_{DDCORE}$ reset) to protect against further software modifications (intentional or unintentional) | – | X |
| | | Digital I/O<br>- Plausibility check | X | – |
| ADCC | | Analog I/O and ADC converter<br>- Plausibility check | X | – |
| ICM (SHA) | Memory and Internal Data Path | All internal and external memories such as QSPI, DDR, and all memories on SMC | X | – |
| NAND Flash Controller ECC | | Non-volatile memory<br>- Mutiple error detection (2 to 32) | – | X |
| System Controller | Supply Monitor | Power supplies<br>- VDDCORE, VDDIO, VDDANA, VDDBU abnormal levels | – | X |
| WDT, RSTC | Watchdog | Watchdog can be fed by an internal always ON clock<br>- Program counter stuck at faults. | X | X |
| | | Watchdog configuration can be locked (write-protected)<br>- Errant writes (Programming errors, errors introduced by system or hardware failures) | – | X |
| | | Watchdog overflow generates a system reset | X | X |
| Cortex MMU | Memory Management Unit | Cortex-A5 Memory Management Unit | – | X |
| MATRIX, AIC, RTC, SYSC, RXLP, ACC, PMC, PIO, MPDDRC, SMC, CLASSD, SSC, TWI, UART, SPI, FLEXCOM, QSPI, TC, PDMIC, ADC | Peripherals | Configuration, Interrupt Enable/Disable, Control registers can be independently write-protected<br>- Errant writes (Programming errors, errors introduced by system or hardware failures) | – | X |

.....continued

| Peripheral | Component | Fault/Error/Feature | Requirements for Class B IEC 60730(1) | General Safety |
|---|---|---|---|---|
| PWM, PIO | PWM | Fault inputs can be configured to put the PWM outputs in Safe mode<br>- Programming errors, errors introduced by system or hardware failures | – | X |
| | | PIO controller can lock the PWM I/O<br>- Programming errors, errors introduced by system or hardware failures | – | X |
| | | Fault inputs can be external (IO) or internal (ADC, TIMER, ACC, etc.)<br>- Programming errors, errors introduced by system or hardware failures | – | X |

**Note:**
Class B IEC 60730 Requirements. Annex H - Table H.1 (H.11.12.7 of edition 3).

## 5.4 Security Features

**Table 5-2. Security Features**

| Peripheral | Function | Description | Comments |
|---|---|---|---|
| TrustZone | Security Enclave | Partition secure/non-secure world | Arm technology |
| Cortex MMU | Memory Management Unit | Cortex-A5 Memory Management Unit | – |
| PIO | I/O Control/ Peripheral Access | When a peripheral is not selected (PIO-controlled), I/O lines have no access to the peripheral. | – |
| | Freeze | Capability to freeze either the functional part or the physical part of the configuration. | Once the freeze command is issued, no modifications to the current configuration are possible. Only a hardware reset allows a change to the configuration. |

.....continued

| Peripheral | Function | Description | Comments |
|---|---|---|---|
| Classical Advanced Software Crypto LIbrary (CASCL) | Cryptography | Software ECC (Asymmetric key algorithm, elliptic curves) | Software library(1) |
| | | Software RSA (Asymmetric key algorithm) | |
| TDES, TRNG | | Hardware-accelerated Triple DES | FIPS-compliant(3) |
| | | True Random Number Generator | |
| AES, SHA | | Hardware-accelerated AES up to 256 bits | |
| | | SHA up to 512 and HMAC-SHA | |
| | Secure Boot | Code encrypted/decrypted, Trusted Code Authentication | Hardware SHA (HMAC) + Software RSA or AES Hardware (CMAC) |
| AESB | AES on-the-fly | On-the-fly encryption/decryption for DDR and QSPI memories | AES128 |
| Memories | Scrambling | On-the-fly scrambling/unscrambling for memories | All internal and external memories such as QSPI, DDR, and all memories on SMC |
| ICM | Memory Integrity Check Monitoring | Uses a hardware Secure Hash Algorithm (up to SHA256) | More robust than CRC. All internal and external memories such as QSPI, DDR, and all memories on SMC can be monitored |
| SECUMOD | JTAG | JTAG entry monitor | These tamper pins (JTAG, test, PIOBUs, monitors, etc.) can be configured to immediately erase Backup memories (BUSRAM4KB and BUREG256b), or generate an interrupt or a wakeup signal. |
| | Test | Test entry monitor | |
| | Active Shield(2) | Die Active Shield | |
| | Voltage Monitoring(2) | VDDBU monitoring | |
| | | VDDCORE monitoring | |
| | Temperature Monitoring(2) | Temperature monitoring | |
| | Frequency Monitoring(2) | 32.768 kHz crystal oscillator monitoring | |
| | | CPU clock monitoring | |
| | IO Tamper Pin | 8 tamper detection pins. Active and Dynamic modes supported. | |
| | Secure Backup SRAM | 5 Kbytes scrambled and non-imprinting avoiding data persistance | 4 Kbytes erasable on tamper detection |
| | Secure Backup Registers | 256-bit register bank, scrambled | Erasable on tamper detection |

.....continued

| Peripheral | Function | Description | Comments |
|---|---|---|---|
| RTC | RTC | Timestamping of tamper events. Protection against bad configuration (invalid entry for date and time are impossible) | All events are logged in the RTC. Timestamping gives the source of the reset/erase memory/ interruption |
| | | RTC robustness against glitch attack on 32 kHz crystal oscillator | – |
| Secure Fuse | JTAG Access Control | Disable JTAG access by fuse bit | – |
| | Secure Debug Disable | JTAG debug allowed in Normal mode only, not in Secure mode | TrustZone |

**Note:**

1. A PCI-certified Advanced Software Crypto Library (ASCL) is available under NDA.
2. Available on SAMA5D23 and SAMA5D28 only. For environmental monitors, refer to *SAMA5D23 and SAMA5D28 Environmental Monitors* (document no. 44036), available under Non-Disclosure Agreement (NDA). Contact a Microchip sales representative for details.
3. Refer to the sections on each peripheral for details on FIPS compliancy.

# 6. Package and Pinout

## 6.1 Packages

The SAMA5D2 is available in the packages listed below.

**Table 6-1. SAMA5D2 Packages**

| Package Name | Pin Count | Ball Pitch |
|---|---|---|
| LFBGA289 | 289 | 0.8 mm |
| TFBGA256 | 256 | 0.4 mm |
| TFBGA196 | 196 | 0.75 mm |

The package mechanical characteristics are described in the section Mechanical Characteristics.

## 6.2 Pinouts

Pinouts are provided in the tables below:

- Pin Description (all packages)
- Pin Description (SAMA5D23 pins different from those in table ”Pin Description (all packages)” )
- Pin Description (SAMA5D28B/C pins different from those in the table ”Pin Description (all packages)” )

I/Os for each peripheral are grouped into IO sets, listed in the column ‘IO Set’ in the pinout tables below. For all peripherals, it is mandatory to use I/Os that belong to the same IO set. The timings are not guaranteed when IOs from different IO sets are mixed.

**Table 6-2. Pin Description (all packages)**

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| U11 | R10 | – | VDDSDMMC | GPIO_EMMC | PA0 | I/O | – | – | A | SDMMC0_CK | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | QSPI0_SCK | O | 1 | |
| | | | | | | | | | F | D0 | I/O | 2 | |
| P10 | R9 | – | VDDSDMMC | GPIO_EMMC | PA1 | I/O | – | – | A | SDMMC0_CMD | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | QSPI0_CS | O | 1 | |
| | | | | | | | | | F | D1 | I/O | 2 | |
| T11 | U11 | – | VDDSDMMC | GPIO_EMMC | PA2 | I/O | – | – | A | SDMMC0_DAT0 | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | QSPI0_IO0 | I/O | 1 | |
| | | | | | | | | | F | D2 | I/O | 2 | |
| R10 | P10 | – | VDDSDMMC | GPIO_EMMC | PA3 | I/O | – | – | A | SDMMC0_DAT1 | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | QSPI0_IO1 | I/O | 1 | |
| | | | | | | | | | F | D3 | I/O | 2 | |
| U12 | P11 | – | VDDSDMMC | GPIO_EMMC | PA4 | I/O | – | – | A | SDMMC0_DAT2 | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | QSPI0_IO2 | I/O | 1 | |
| | | | | | | | | | F | D4 | I/O | 2 | |
| T12 | V11 | – | VDDSDMMC | GGPIO_EMMC | PA5 | I/O | – | – | A | SDMMC0_DAT3 | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | QSPI0_IO3 | I/O | 1 | |
| | | | | | | | | | F | D5 | I/O | 2 | |
| R12 | U12 | – | VDDSDMMC | GPIO_EMMC | PA6 | I/O | – | – | A | SDMMC0_DAT4 | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | QSPI1_SCK | O | 1 | |
| | | | | | | | | | D | TIOA5 | I/O | 1 | |
| | | | | | | | | | E | FLEXCOM2_IO0 | I/O | 1 | |
| | | | | | | | | | F | D6 | I/O | 2 | |
| T13 | V12 | – | VDDSDMMC | GPIO_EMMC | PA7 | I/O | – | – | A | SDMMC0_DAT5 | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | QSPI1_IO0 | I/O | 1 | |
| | | | | | | | | | D | TIOB5 | I/O | 1 | |
| | | | | | | | | | E | FLEXCOM2_IO1 | I/O | 1 | |
| | | | | | | | | | F | D7 | I/O | 2 | |
| N10 | N11 | – | VDDSDMMC | GPIO_EMMC | PA8 | I/O | – | – | A | SDMMC0_DAT6 | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | QSPI1_IO1 | I/O | 1 | |
| | | | | | | | | | D | TCLK5 | I | 1 | |
| | | | | | | | | | E | FLEXCOM2_IO2 | I/O | 1 | |
| | | | | | | | | | F | NWE/NANDWE | O | 2 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| N11 | P12 | – | VDDSDMMC | GPIO_EMMC | PA9 | I/O | – | – | A | SDMMC0_DAT7 | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | QSPI1_IO2 | I/O | 1 | |
| | | | | | | | | | D | TIOA4 | I/O | 1 | |
| | | | | | | | | | E | FLEXCOM2_IO3 | O | 1 | |
| | | | | | | | | | F | NCS3 | O | 2 | |
| U13 | U13 | – | VDDSDMMC | GPIO_EMMC | PA10 | I/O | – | – | A | SDMMC0_RSTN | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | QSPI1_IO3 | I/O | 1 | |
| | | | | | | | | | D | TIOB4 | I/O | 1 | |
| | | | | | | | | | E | FLEXCOM2_IO4 | O | 1 | |
| | | | | | | | | | F | A21/NANDALE | O | 2 | |
| P15 | R14 | – | VDDIOP1 | GPIO | PA11 | I/O | – | – | A | SDMMC0_1V8SEL | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | QSPI1_CS | O | 1 | |
| | | | | | | | | | D | TCLK4 | I | 1 | |
| | | | | | | | | | F | A22/NANDCLE | O | 2 | |
| N15 | N13 | – | VDDIOP1 | GPIO | PA12 | I/O | – | – | A | SDMMC0_WP | I | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | IRQ | I | 1 | |
| | | | | | | | | | F | NRD/NANDOE | O | 2 | |
| P16 | P14 | – | VDDIOP1 | GPIO | PA13 | I/O | – | – | A | SDMMC0_CD | I | 1 | PIO, I, PU, ST |
| | | | | | | | | | E | FLEXCOM3_IO1 | I/O | 1 | |
| | | | | | | | | | F | D8 | I/O | 2 | |
| M14 | P17 | – | VDDIOP1 | GPIO_QSPI | PA14 | I/O | – | – | A | SPI0_SPCK | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | TK1 | I/O | 1 | |
| | | | | | | | | | C | QSPI0_SCK | O | 2 | |
| | | | | | | | | | D | I2SC1_MCK | O | 2 | |
| | | | | | | | | | E | FLEXCOM3_IO2 | I/O | 1 | |
| | | | | | | | | | F | D9 | I/O | 2 | |
| N16 | R18 | – | VDDIOP1 | GPIO | PA15 | I/O | – | – | A | SPI0_MOSI | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | TF1 | I/O | 1 | |
| | | | | | | | | | C | QSPI0_CS | O | 2 | |
| | | | | | | | | | D | I2SC1_CK | I/O | 2 | |
| | | | | | | | | | E | FLEXCOM3_IO0 | I/O | 1 | |
| | | | | | | | | | F | D10 | I/O | 2 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| M10 | N15 | – | VDDIOP1 | GPIO_IO | PA16 | I/O | – | – | A | SPI0_MISO | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | TD1 | O | 1 | |
| | | | | | | | | | C | QSPI0_IO0 | I/O | 2 | |
| | | | | | | | | | D | I2SC1_WS | I/O | 2 | |
| | | | | | | | | | E | FLEXCOM3_IO3 | O | 1 | |
| | | | | | | | | | F | D11 | I/O | 2 | |
| N17 | P18 | – | VDDIOP1 | GPIO_IO | PA17 | I/O | – | – | A | SPI0_NPCS0 | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | RD1 | I | 1 | |
| | | | | | | | | | C | QSPI0_IO1 | I/O | 2 | |
| | | | | | | | | | D | I2SC1_DI0 | I | 2 | |
| | | | | | | | | | E | FLEXCOM3_IO4 | O | 1 | |
| | | | | | | | | | F | D12 | I/O | 2 | |
| U14 | M9 | L9 | VDDIOP1 | GPIO_IO | PA18 | I/O | – | – | A | SPI0_NPCS1 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | RK1 | I/O | 1 | |
| | | | | | | | | | C | QSPI0_IO2 | I/O | 2 | |
| | | | | | | | | | D | I2SC1_DO0 | O | 2 | |
| | | | | | | | | | E | SDMMC1_DAT0 | I/O | 1 | |
| | | | | | | | | | F | D13 | I/O | 2 | |
| T14 | V13 | N9 | VDDIOP1 | GPIO_IO | PA19 | I/O | – | – | A | SPI0_NPCS2 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | RF1 | I/O | 1 | |
| | | | | | | | | | C | QSPI0_IO3 | I/O | 2 | |
| | | | | | | | | | D | TIOA0 | I/O | 1 | |
| | | | | | | | | | E | SDMMC1_DAT1 | I/O | 1 | |
| | | | | | | | | | F | D14 | I/O | 2 | |
| P12 | L9 | M9 | VDDIOP1 | GPIO_IO | PA20 | I/O | – | – | A | SPI0_NPCS3 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | D | TIOB0 | I/O | 1 | |
| | | | | | | | | | E | SDMMC1_DAT2 | I/O | 1 | |
| | | | | | | | | | F | D15 | I/O | 2 | |
| R13 | M10 | M10 | VDDIOP1 | GPIO_IO | PA21 | I/O | – | – | A | IRQ | I | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | PCK2 | O | 3 | |
| | | | | | | | | | D | TCLK0 | I | 1 | |
| | | | | | | | | | E | SDMMC1_DAT3 | I/O | 1 | |
| | | | | | | | | | F | NANDRDY | I | 2 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| U15 | V14 | P9 | VDDIOP1 | GPIO_QSPI | PA22 | I/O | – | – | A | FLEXCOM1_IO2 | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | D0 | I/O | 1 | |
| | | | | | | | | | C | TCK | I | 4 | |
| | | | | | | | | | D | SPI1_SPCK | I/O | 2 | |
| | | | | | | | | | E | SDMMC1_CK | I/O | 1 | |
| | | | | | | | | | F | QSPI0_SCK | O | 3 | |
| U16 | U14 | P10 | VDDIOP1 | GPIO | PA23 | I/O | – | – | A | FLEXCOM1_IO1 | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | D1 | I/O | 1 | |
| | | | | | | | | | C | TDI | I | 4 | |
| | | | | | | | | | D | SPI1_MOSI | I/O | 2 | |
| | | | | | | | | | F | QSPI0_CS | O | 3 | |
| T15 | R13 | N10 | VDDIOP1 | GPIO_IO | PA24 | I/O | – | – | A | FLEXCOM1_IO0 | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | D2 | I/O | 1 | |
| | | | | | | | | | C | TDO | O | 4 | |
| | | | | | | | | | D | SPI1_MISO | I/O | 2 | |
| | | | | | | | | | F | QSPI0_IO0 | I/O | 3 | |
| U17 | U15 | L10 | VDDIOP1 | GPIO_IO | PA25 | I/O | – | – | A | FLEXCOM1_IO3 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | D3 | I/O | 1 | |
| | | | | | | | | | C | TMS | I | 4 | |
| | | | | | | | | | D | SPI1_NPCS0 | I/O | 2 | |
| | | | | | | | | | F | QSPI0_IO1 | I/O | 3 | |
| P13 | L10 | P11 | VDDIOP1 | GPIO_IO | PA26 | I/O | – | – | A | FLEXCOM1_IO4 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | D4 | I/O | 1 | |
| | | | | | | | | | C | NTRST | I | 4 | |
| | | | | | | | | | D | SPI1_NPCS1 | O | 2 | |
| | | | | | | | | | F | QSPI0_IO2 | I/O | 3 | |
| T16 | V17 | P12 | VDDIOP1 | GPIO_IO | PA27 | I/O | – | – | A | TIOA1 | I/O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | D5 | I/O | 1 | |
| | | | | | | | | | C | SPI0_NPCS2 | O | 2 | |
| | | | | | | | | | D | SPI1_NPCS2 | O | 2 | |
| | | | | | | | | | E | SDMMC1_RSTN | O | 1 | |
| | | | | | | | | | F | QSPI0_IO3 | I/O | 3 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| R16 | U16 | M11 | VDDIOP1 | GPIO | PA28 | I/O | – | – | A | TIOB1 | I/O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | D6 | I/O | 1 | |
| | | | | | | | | | C | SPI0_NPCS3 | O | 2 | |
| | | | | | | | | | D | SPI1_NPCS3 | O | 2 | |
| | | | | | | | | | E | SDMMC1_CMD | I/O | 1 | |
| | | | | | | | | | F | CLASSD_L0 | O | 1 | |
| T17 | U17 | N11 | VDDIOP1 | GPIO | PA29 | I/O | – | – | A | TCLK1 | I | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | D7 | I/O | 1 | |
| | | | | | | | | | C | SPI0_NPCS1 | O | 2 | |
| | | | | | | | | | E | SDMMC1_WP | I | 1 | |
| | | | | | | | | | F | CLASSD_L1 | O | 1 | |
| R15 | V18 | N12 | VDDIOP1 | GPIO | PA30 | I/O | – | – | B | NWE/NANDWE | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | C | SPI0_NPCS0 | I/O | 2 | |
| | | | | | | | | | D | PWMH0 | O | 1 | |
| | | | | | | | | | E | SDMMC1_CD | I | 1 | |
| | | | | | | | | | F | CLASSD_L2 | O | 1 | |
| R17 | U18 | M12 | VDDIOP1 | GPIO | PA31 | I/O | – | – | B | NCS3 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | C | SPI0_MISO | I/O | 2 | |
| | | | | | | | | | D | PWML0 | O | 1 | |
| | | | | | | | | | F | CLASSD_L3 | O | 1 | |
| J8 | G9 | A6 | VDDIOP0 | GPIO | PB0 | I/O | – | – | B | A21/NANDALE | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | C | SPI0_MOSI | I/O | 2 | |
| | | | | | | | | | D | PWMH1 | O | 1 | |
| A8 | A7 | A5 | VDDIOP0 | GPIO | PB1 | I/O | – | – | B | A22/NANDCLE | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | C | SPI0_SPCK | I/O | 2 | |
| | | | | | | | | | D | PWML1 | O | 1 | |
| | | | | | | | | | F | CLASSD_R0 | O | 1 | |
| A7 | B7 | B6 | VDDIOP0 | GPIO | PB2 | I/O | – | – | B | NRD/NANDOE | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | D | PWMFI0 | I | 1 | |
| | | | | | | | | | F | CLASSD_R1 | O | 1 | |
| A6 | B6 | B5 | VDDIOP0 | GPIO | PB3 | I/O | – | – | A | URXD4 | I | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | D8 | I/O | 1 | |
| | | | | | | | | | C | IRQ | I | 3 | |
| | | | | | | | | | D | PWMEXTRG1 | I | 1 | |
| | | | | | | | | | F | CLASSD_R2 | O | 1 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| B6 | A6 | A4 | VDDIOP0 | GPIO | PB4 | I/O | – | – | A | UTXD4 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | D9 | I/O | 1 | |
| | | | | | | | | | C | FIQ | I | 4 | |
| | | | | | | | | | F | CLASSD_R3 | O | 1 | |
| B7 | D7 | D6 | VDDIOP0 | GPIO_QSPI | PB5 | I/O | – | – | A | TCLK2 | I | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | D10 | I/O | 1 | |
| | | | | | | | | | C | PWMH2 | O | 1 | |
| | | | | | | | | | D | QSPI1_SCK | O | 2 | |
| | | | | | | | | | F | GTSUCOMP | O | 3 | |
| C7 | B5 | A3 | VDDIOP0 | GPIO | PB6 | I/O | – | – | A | TIOA2 | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | D11 | I/O | 1 | |
| | | | | | | | | | C | PWML2 | O | 1 | |
| | | | | | | | | | D | QSPI1_CS | O | 2 | |
| | | | | | | | | | F | GTXER | O | 3 | |
| C6 | A5 | B4 | VDDIOP0 | GPIO_IO | PB7 | I/O | – | – | A | TIOB2 | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | D12 | I/O | 1 | |
| | | | | | | | | | C | PWMH3 | O | 1 | |
| | | | | | | | | | D | QSPI1_IO0 | I/O | 2 | |
| | | | | | | | | | F | GRXCK | I | 3 | |
| A5 | E7 | A2 | VDDIOP0 | GPIO_IO | PB8 | I/O | – | – | A | TCLK3 | I | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | D13 | I/O | 1 | |
| | | | | | | | | | C | PWML3 | O | 1 | |
| | | | | | | | | | D | QSPI1_IO1 | I/O | 2 | |
| | | | | | | | | | F | GCRS | I | 3 | |
| A4 | F6 | B3 | VDDIOP0 | GPIO_IO | PB9 | I/O | – | – | A | TIOA3 | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | D14 | I/O | 1 | |
| | | | | | | | | | C | PWMFI1 | I | 1 | |
| | | | | | | | | | D | QSPI1_IO2 | I/O | 2 | |
| | | | | | | | | | F | GCOL | I | 3 | |
| H8 | D6 | A1 | VDDIOP0 | GPIO_IO | PB10 | I/O | – | – | A | TIOB3 | I/O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | D15 | I/O | 1 | |
| | | | | | | | | | C | PWMEXTRG2 | I | 1 | |
| | | | | | | | | | D | QSPI1_IO3 | I/O | 2 | |
| | | | | | | | | | F | GRX2 | I | 3 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| B5 | A4 | B1 | VDDIOP0 | GPIO | PB11 | I/O | – | – | A | LCDDAT0 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A0/NBS0 | O | 1 | |
| | | | | | | | | | C | URXD3 | I | 3 | |
| | | | | | | | | | D | PDMIC_DAT | | 2 | |
| | | | | | | | | | F | GRX3 | I | 3 | |
| D6 | B3 | B2 | VDDIOP0 | GPIO | PB12 | I/O | – | – | A | LCDDAT1 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A1 | O | 1 | |
| | | | | | | | | | C | UTXD3 | O | 3 | |
| | | | | | | | | | D | PDMIC_CLK | | 2 | |
| | | | | | | | | | F | GTX2 | O | 3 | |
| B4 | A3 | C1 | VDDIOP0 | GPIO | PB13 | I/O | – | – | A | LCDDAT2 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A2 | O | 1 | |
| | | | | | | | | | C | PCK1 | O | 3 | |
| | | | | | | | | | F | GTX3 | O | 3 | |
| C5 | B4 | D5 | VDDIOP0 | GPIO_QSPI | PB14 | I/O | – | – | A | LCDDAT3 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A3 | O | 1 | |
| | | | | | | | | | C | TK1 | I/O | 2 | |
| | | | | | | | | | D | I2SC1_MCK | O | 1 | |
| | | | | | | | | | E | QSPI1_SCK | O | 3 | |
| | | | | | | | | | F | GTXCK | I/O | 3 | |
| H7 | G8 | E5 | VDDIOP0 | GPIO | PB15 | I/O | – | – | A | LCDDAT4 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A4 | O | 1 | |
| | | | | | | | | | C | TF1 | I/O | 2 | |
| | | | | | | | | | D | I2SC1_CK | I/O | 1 | |
| | | | | | | | | | E | QSPI1_CS | O | 3 | |
| | | | | | | | | | F | GTXEN | O | 3 | |
| D5 | E5 | C5 | VDDIOP0 | GPIO_IO | PB16 | I/O | – | – | A | LCDDAT5 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A5 | O | 1 | |
| | | | | | | | | | C | TD1 | O | 2 | |
| | | | | | | | | | D | I2SC1_WS | I/O | 1 | |
| | | | | | | | | | E | QSPI1_IO0 | I/O | 3 | |
| | | | | | | | | | F | GRXDV | I | 3 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| C4 | G7 | C2 | VDDIOP0 | GPIO_IO | PB17 | I/O | – | – | A | LCDDAT6 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A6 | O | 1 | |
| | | | | | | | | | C | RD1 | I | 2 | |
| | | | | | | | | | D | I2SC1_DI0 | I | 1 | |
| | | | | | | | | | E | QSPI1_IO1 | I/O | 3 | |
| | | | | | | | | | F | GRXER | I | 3 | |
| A3 | A2 | D4 | VDDIOP0 | GPIO_IO | PB18 | I/O | – | – | A | LCDDAT7 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A7 | O | 1 | |
| | | | | | | | | | C | RK1 | I/O | 2 | |
| | | | | | | | | | D | I2SC1_DO0 | O | 1 | |
| | | | | | | | | | E | QSPI1_IO2 | I/O | 3 | |
| | | | | | | | | | F | GRX0 | I | 3 | |
| D4 | H7 | C4 | VDDIOP0 | GPIO_IO | PB19 | I/O | – | – | A | LCDDAT8 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A8 | O | 1 | |
| | | | | | | | | | C | RF1 | I/O | 2 | |
| | | | | | | | | | D | TIOA3 | I/O | 2 | |
| | | | | | | | | | E | QSPI1_IO3 | I/O | 3 | |
| | | | | | | | | | F | GRX1 | I | 3 | |
| B3 | A1 | C3 | VDDIOP0 | GPIO | PB20 | I/O | – | – | A | LCDDAT9 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A9 | O | 1 | |
| | | | | | | | | | C | TK0 | I/O | 1 | |
| | | | | | | | | | D | TIOB3 | I/O | 2 | |
| | | | | | | | | | E | PCK1 | O | 4 | |
| | | | | | | | | | F | GTX0 | O | 3 | |
| A2 | D2 | D1 | VDDIOP0 | GPIO | PB21 | I/O | – | – | A | LCDDAT10 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A10 | O | 1 | |
| | | | | | | | | | C | TF0 | I/O | 1 | |
| | | | | | | | | | D | TCLK3 | I | 2 | |
| | | | | | | | | | E | FLEXCOM3_IO2 | I/O | 3 | |
| | | | | | | | | | F | GTX1 | O | 3 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| C3 | G5 | D2 | VDDIOP0 | GPIO | PB22 | I/O | – | – | A | LCDDAT11 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A11 | O | 1 | |
| | | | | | | | | | C | TD0 | O | 1 | |
| | | | | | | | | | D | TIOA2 | I/O | 2 | |
| | | | | | | | | | E | FLEXCOM3_IO1 | I/O | 3 | |
| | | | | | | | | | F | GMDC | O | 3 | |
| A1 | C2 | E1 | VDDIOP0 | GPIO | PB23 | I/O | – | – | A | LCDDAT12 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A12 | O | 1 | |
| | | | | | | | | | C | RD0 | I | 1 | |
| | | | | | | | | | D | TIOB2 | I/O | 2 | |
| | | | | | | | | | E | FLEXCOM3_IO0 | I/O | 3 | |
| | | | | | | | | | F | GMDIO | I/O | 3 | |
| E5 | F4 | D3 | VDDIOP0 | GPIO | PB24 | I/O | – | – | A | LCDDAT13 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A13 | O | 1 | |
| | | | | | | | | | C | RK0 | I/O | 1 | |
| | | | | | | | | | D | TCLK2 | I | 2 | |
| | | | | | | | | | E | FLEXCOM3_IO3 | O | 3 | |
| | | | | | | | | | F | ISC_D10 | I | 3 | |
| B2 | C1 | E3 | VDDIOP0 | GPIO | PB25 | I/O | – | – | A | LCDDAT14 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A14 | O | 1 | |
| | | | | | | | | | C | RF0 | I/O | 1 | |
| | | | | | | | | | E | FLEXCOM3_IO4 | O | 3 | |
| | | | | | | | | | F | ISC_D11 | I | 3 | |
| E4 | E4 | E2 | VDDIOP0 | GPIO | PB26 | I/O | – | – | A | LCDDAT15 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A15 | O | 1 | |
| | | | | | | | | | C | URXD0 | I | 1 | |
| | | | | | | | | | D | PDMIC_DAT | | 1 | |
| | | | | | | | | | F | ISC_D0 | I | 3 | |
| B1 | F1 | E6 | VDDIOP0 | GPIO | PB27 | I/O | – | – | A | LCDDAT16 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A16 | O | 1 | |
| | | | | | | | | | C | UTXD0 | O | 1 | |
| | | | | | | | | | D | PDMIC_CLK | | 1 | |
| | | | | | | | | | F | ISC_D1 | I | 3 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| C2 | D1 | F1 | VDDIOP0 | GPIO | PB28 | I/O | – | – | A | LCDDAT17 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A17 | O | 1 | |
| | | | | | | | | | C | FLEXCOM0_IO0 | I/O | 1 | |
| | | | | | | | | | D | TIOA5 | I/O | 2 | |
| | | | | | | | | | F | ISC_D2 | I | 3 | |
| D3 | F2 | F6 | VDDIOP0 | GPIO | PB29 | I/O | – | – | A | LCDDAT18 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A18 | O | 1 | |
| | | | | | | | | | C | FLEXCOM0_IO1 | I/O | 1 | |
| | | | | | | | | | D | TIOB5 | I/O | 2 | |
| | | | | | | | | | F | ISC_D3 | I | 3 | |
| D2 | E2 | F2 | VDDIOP0 | GPIO | PB30 | I/O | – | – | A | LCDDAT19 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A19 | O | 1 | |
| | | | | | | | | | C | FLEXCOM0_IO2 | I/O | 1 | |
| | | | | | | | | | D | TCLK5 | I | 2 | |
| | | | | | | | | | F | ISC_D4 | I | 3 | |
| C1 | E1 | F7 | VDDIOP0 | GPIO | PB31 | I/O | – | – | A | LCDDAT20 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A20 | O | 1 | |
| | | | | | | | | | C | FLEXCOM0_IO3 | O | 1 | |
| | | | | | | | | | D | TWD0 | I/O | 1 | |
| | | | | | | | | | F | ISC_D5 | I | 3 | |
| P17 | R15 | M13 | VDDIOP1 | GPIO | PC0 | I/O | – | – | A | LCDDAT21 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A23 | O | 1 | |
| | | | | | | | | | C | FLEXCOM0_IO4 | O | 1 | |
| | | | | | | | | | D | TWCK0 | I/O | 1 | |
| | | | | | | | | | F | ISC_D6 | I | 3 | |
| N12 | M11 | P13 | VDDIOP1 | GPIO | PC1 | I/O | – | – | A | LCDDAT22 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A24 | O | 1 | |
| | | | | | | | | | C | CANTX0 | O | 1 | |
| | | | | | | | | | D | SPI1_SPCK | I/O | 1 | |
| | | | | | | | | | E | I2SC0_CK | I/O | 1 | |
| | | | | | | | | | F | ISC_D7 | I | 3 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| N14 | P15 | N13 | VDDIOP1 | GPIO | PC2 | I/O | – | – | A | LCDDAT23 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | A25 | O | 1 | |
| | | | | | | | | | C | CANRX0 | I | 1 | |
| | | | | | | | | | D | SPI1_MOSI | I/O | 1 | |
| | | | | | | | | | E | I2SC0_MCK | O | 1 | |
| | | | | | | | | | F | ISC_D8 | I | 3 | |
| M15 | K9 | K10 | VDDIOP1 | GPIO | PC3 | I/O | – | – | A | LCDPWM | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | NWAIT | I | 1 | |
| | | | | | | | | | C | TIOA1 | I/O | 1 | |
| | | | | | | | | | D | SPI1_MISO | I/O | 1 | |
| | | | | | | | | | E | I2SC0_WS | I/O | 1 | |
| | | | | | | | | | F | ISC_D9 | I | 3 | |
| M11 | K10 | P14 | VDDIOP1 | GPIO | PC4 | I/O | – | – | A | LCDDISP | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | NWR1/NBS1 | O | 1 | |
| | | | | | | | | | C | TIOB1 | I/O | 1 | |
| | | | | | | | | | D | SPI1_NPCS0 | I/O | 1 | |
| | | | | | | | | | E | I2SC0_DI0 | I | 1 | |
| | | | | | | | | | F | ISC_PCK | I | 3 | |
| L10 | L11 | J8 | VDDIOP1 | GPIO | PC5 | I/O | – | – | A | LCDVSYNC | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | NCS0 | O | 1 | |
| | | | | | | | | | C | TCLK1 | I | 1 | |
| | | | | | | | | | D | SPI1_NPCS1 | O | 1 | |
| | | | | | | | | | E | I2SC0_DO0 | O | 1 | |
| | | | | | | | | | F | ISC_VSYNC | I | 3 | |
| K10 | L12 | N14 | VDDIOP1 | GPIO | PC6 | I/O | – | – | A | LCDHSYNC | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | NCS1 | O | 1 | |
| | | | | | | | | | C | TWD1 | I/O | 1 | |
| | | | | | | | | | D | SPI1_NPCS2 | O | 1 | |
| | | | | | | | | | F | ISC_HSYNC | I | 3 | |
| M16 | M12 | M14 | VDDIOP1 | GPIO_CLK | PC7 | I/O | – | – | A | LCDPCK | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | NCS2 | O | 1 | |
| | | | | | | | | | C | TWCK1 | I/O | 1 | |
| | | | | | | | | | D | SPI1_NPCS3 | O | 1 | |
| | | | | | | | | | E | URXD1 | I | 2 | |
| | | | | | | | | | F | ISC_MCK | O | 3 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| J10 | K11 | J9 | VDDIOP1 | GPIO | PC8 | I/O | – | – | A | LCDDEN | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | NANDRDY | I | 1 | |
| | | | | | | | | | C | FIQ | I | 1 | |
| | | | | | | | | | D | PCK0 | O | 3 | |
| | | | | | | | | | E | UTXD1 | O | 2 | |
| | | | | | | | | | F | ISC_FIELD | I | 3 | |
| D1 | – | – | VDDISC | GPIO | PC9 | I/O | – | – | A | FIQ | I | 3 | PIO, I, PU, ST |
| | | | | | | | | | B | GTSUCOMP | O | 1 | |
| | | | | | | | | | C | ISC_D0 | I | 1 | |
| | | | | | | | | | D | TIOA4 | I/O | 2 | |
| E3 | – | – | VDDISC | GPIO | PC10 | I/O | – | – | A | LCDDAT2 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GTXCK | I/O | 1 | |
| | | | | | | | | | C | ISC_D1 | I | 1 | |
| | | | | | | | | | D | TIOB4 | I/O | 2 | |
| | | | | | | | | | E | CANTX0 | O | 2 | |
| E2 | – | – | VDDISC | GPIO | PC11 | I/O | – | – | A | LCDDAT3 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GTXEN | O | 1 | |
| | | | | | | | | | C | ISC_D2 | I | 1 | |
| | | | | | | | | | D | TCLK4 | I | 2 | |
| | | | | | | | | | E | CANRX0 | I | 2 | |
| | | | | | | | | | F | A0/NBS0 | O | 2 | |
| E1 | – | – | VDDISC | GPIO | PC12 | I/O | – | – | A | LCDDAT4 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GRXDV | I | 1 | |
| | | | | | | | | | C | ISC_D3 | I | 1 | |
| | | | | | | | | | D | URXD3 | I | 1 | |
| | | | | | | | | | E | TK0 | I/O | 2 | |
| | | | | | | | | | F | A1 | O | 2 | |
| F3 | – | – | VDDISC | GPIO | PC13 | I/O | – | – | A | LCDDAT5 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GRXER | I | 1 | |
| | | | | | | | | | C | ISC_D4 | I | 1 | |
| | | | | | | | | | D | UTXD3 | O | 1 | |
| | | | | | | | | | E | TF0 | I/O | 2 | |
| | | | | | | | | | F | A2 | O | 2 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| F5 | – | – | VDDISC | GPIO | PC14 | I/O | – | – | A | LCDDAT6 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GRX0 | I | 1 | |
| | | | | | | | | | C | ISC_D5 | I | 1 | |
| | | | | | | | | | E | TD0 | O | 2 | |
| | | | | | | | | | F | A3 | O | 2 | |
| F2 | – | – | VDDISC | GPIO | PC15 | I/O | – | – | A | LCDDAT7 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GRX1 | I | 1 | |
| | | | | | | | | | C | ISC_D6 | I | 1 | |
| | | | | | | | | | E | RD0 | I | 2 | |
| | | | | | | | | | F | A4 | O | 2 | |
| G6 | – | – | VDDISC | GPIO | PC16 | I/O | – | – | A | LCDDAT10 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GTX0 | O | 1 | |
| | | | | | | | | | C | ISC_D7 | I | 1 | |
| | | | | | | | | | E | RK0 | I/O | 2 | |
| | | | | | | | | | F | A5 | O | 2 | |
| F1 | – | – | VDDISC | GPIO | PC17 | I/O | – | – | A | LCDDAT11 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GTX1 | O | 1 | |
| | | | | | | | | | C | ISC_D8 | I | 1 | |
| | | | | | | | | | E | RF0 | I/O | 2 | |
| | | | | | | | | | F | A6 | O | 2 | |
| H6 | – | – | VDDISC | GPIO | PC18 | I/O | – | – | A | LCDDAT12 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GMDC | O | 1 | |
| | | | | | | | | | C | ISC_D9 | I | 1 | |
| | | | | | | | | | E | FLEXCOM3_IO2 | I/O | 2 | |
| | | | | | | | | | F | A7 | O | 2 | |
| G2 | – | – | VDDISC | GPIO | PC19 | I/O | – | – | A | LCDDAT13 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GMDIO | I/O | 1 | |
| | | | | | | | | | C | ISC_D10 | I | 1 | |
| | | | | | | | | | E | FLEXCOM3_IO1 | I/O | 2 | |
| | | | | | | | | | F | A8 | O | 2 | |
| G3 | – | – | VDDISC | GPIO | PC20 | I/O | – | – | A | LCDDAT14 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GRXCK | I | 1 | |
| | | | | | | | | | C | ISC_D11 | I | 1 | |
| | | | | | | | | | E | FLEXCOM3_IO0 | I/O | 2 | |
| | | | | | | | | | F | A9 | O | 2 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| G1 | – | – | VDDISC | GPIO | PC21 | I/O | – | – | A | LCDDAT15 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GTXER | O | 1 | |
| | | | | | | | | | C | ISC_PCK | I | 1 | |
| | | | | | | | | | E | FLEXCOM3_IO3 | O | 2 | |
| | | | | | | | | | F | A10 | O | 2 | |
| H2 | – | – | VDDISC | GPIO | PC22 | I/O | – | – | A | LCDDAT18 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GCRS | I | 1 | |
| | | | | | | | | | C | ISC_VSYNC | I | 1 | |
| | | | | | | | | | E | FLEXCOM3_IO4 | O | 2 | |
| | | | | | | | | | F | A11 | O | 2 | |
| G5 | – | – | VDDISC | GPIO | PC23 | I/O | – | – | A | LCDDAT19 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GCOL | I | 1 | |
| | | | | | | | | | C | ISC_HSYNC | I | 1 | |
| | | | | | | | | | F | A12 | O | 2 | |
| H1 | – | – | VDDISC | GPIO_CLK | PC24 | I/O | – | – | A | LCDDAT20 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GRX2 | I | 1 | |
| | | | | | | | | | C | ISC_MCK | O | 1 | |
| | | | | | | | | | F | A13 | O | 2 | |
| H5 | – | – | VDDISC | GPIO | PC25 | I/O | – | – | A | LCDDAT21 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GRX3 | I | 1 | |
| | | | | | | | | | C | ISC_FIELD | I | 1 | |
| | | | | | | | | | F | A14 | O | 2 | |
| J9 | – | – | VDDIOP2 | GPIO | PC26 | I/O | – | – | A | LCDDAT22 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GTX2 | O | 1 | |
| | | | | | | | | | D | CANTX1 | O | 1 | |
| | | | | | | | | | F | A15 | O | 2 | |
| H9 | – | – | VDDIOP2 | GPIO | PC27 | I/O | – | – | A | LCDDAT23 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | GTX3 | O | 1 | |
| | | | | | | | | | C | PCK1 | O | 2 | |
| | | | | | | | | | D | CANRX1 | I | 1 | |
| | | | | | | | | | E | TWD0 | I/O | 2 | |
| | | | | | | | | | F | A16 | O | 2 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| E8 | – | – | VDDIOP2 | GPIO | PC28 | I/O | – | – | A | LCDPWM | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | FLEXCOM4_IO0 | I/O | 1 | |
| | | | | | | | | | C | PCK2 | O | 1 | |
| | | | | | | | | | E | TWCK0 | I/O | 2 | |
| | | | | | | | | | F | A17 | O | 2 | |
| G8 | – | – | VDDIOP2 | GPIO | PC29 | I/O | – | – | A | LCDDISP | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | FLEXCOM4_IO1 | I/O | 1 | |
| | | | | | | | | | F | A18 | O | 2 | |
| F8 | – | – | VDDIOP2 | GPIO | PC30 | I/O | – | – | A | LCDVSYNC | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | FLEXCOM4_IO2 | I/O | 1 | |
| | | | | | | | | | F | A19 | O | 2 | |
| D8 | – | – | VDDIOP2 | GPIO | PC31 | I/O | – | – | A | LCDHSYNC | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | FLEXCOM4_IO3 | O | 1 | |
| | | | | | | | | | C | URXD3 | I | 2 | |
| | | | | | | | | | F | A20 | O | 2 | |
| G10 | E9 | – | VDDIOP2 | GPIO_CLK | PD0 | I/O | – | – | A | LCDPCK | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | FLEXCOM4_IO4 | O | 1 | |
| | | | | | | | | | C | UTXD3 | O | 2 | |
| | | | | | | | | | D | GTSUCOMP | O | 2 | |
| | | | | | | | | | F | A23 | O | 2 | |
| E10 | F8 | – | VDDIOP2 | GPIO | PD1 | I/O | – | – | A | LCDDEN | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | D | GRXCK | I | 2 | |
| | | | | | | | | | F | A24 | O | 2 | |
| G9 | F9 | – | VDDIOP2 | GPIO_CLK | PD2 | I/O | – | – | A | URXD1 | I | 1 | PIO, I, PU, ST |
| | | | | | | | | | D | GTXER | O | 2 | |
| | | | | | | | | | E | ISC_MCK | O | 2 | |
| | | | | | | | | | F | A25 | O | 2 | |
| K1 | J4 | – | VDDANA | GPIO_AD | PD3 | I/O | PTC_X0 | – | A | UTXD1 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | FIQ | I | 2 | |
| | | | | | | | | | D | GCRS | I | 2 | |
| | | | | | | | | | E | ISC_D11 | I | 2 | |
| | | | | | | | | | F | NWAIT | I | 2 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| J6 | H6 | – | VDDANA | GPIO_AD | PD4 | I/O | PTC_X1 | – | A | TWD1 | I/O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | URXD2 | I | 1 | |
| | | | | | | | | | D | GCOL | I | 2 | |
| | | | | | | | | | E | ISC_D10 | I | 2 | |
| | | | | | | | | | F | NCS0 | O | 2 | |
| J4 | H1 | – | VDDANA | GPIO_AD | PD5 | I/O | PTC_X2 | – | A | TWCK1 | I/O | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | UTXD2 | O | 1 | |
| | | | | | | | | | D | GRX2 | I | 2 | |
| | | | | | | | | | E | ISC_D9 | I | 2 | |
| | | | | | | | | | F | NCS1 | O | 2 | |
| J2 | G4 | – | VDDANA | GPIO_AD | PD6 | I/O | PTC_X3 | – | A | TCK | I | 2 | PIO, I, PU, ST |
| | | | | | | | | | B | PCK1 | O | 1 | |
| | | | | | | | | | D | GRX3 | I | 2 | |
| | | | | | | | | | E | ISC_D8 | I | 2 | |
| | | | | | | | | | F | NCS2 | O | 2 | |
| J7 | H5 | F5 | VDDANA | GPIO_AD | PD7 | I/O | PTC_X4 | – | A | TDI | I | 2 | PIO, I, PU, ST |
| | | | | | | | | | C | UTMI_RXVAL | O | 1 | |
| | | | | | | | | | D | GTX2 | O | 2 | |
| | | | | | | | | | E | ISC_D0 | I | 2 | |
| | | | | | | | | | F | NWR1/NBS1 | O | 2 | |
| J1 | G1 | F3 | VDDANA | GPIO_AD | PD8 | I/O | PTC_X5 | – | A | TDO | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | C | UTMI_RXERR | O | 1 | |
| | | | | | | | | | D | GTX3 | O | 2 | |
| | | | | | | | | | E | ISC_D1 | I | 2 | |
| | | | | | | | | | F | NANDRDY | I | 2 | |
| K9 | H4 | G5 | VDDANA | GPIO_AD | PD9 | I/O | PTC_X6 | – | A | TMS | I | 2 | PIO, I, PU, ST |
| | | | | | | | | | C | UTMI_RXACT | O | 1 | |
| | | | | | | | | | D | GTXCK | I/O | 2 | |
| | | | | | | | | | E | ISC_D2 | I | 2 | |
| J3 | G2 | G4 | VDDANA | GPIO_AD | PD10 | I/O | PTC_X7 | – | A | NTRST | I | 2 | PIO, I, PU, ST |
| | | | | | | | | | C | UTMI_HDIS | O | 1 | |
| | | | | | | | | | D | GTXEN | O | 2 | |
| | | | | | | | | | E | ISC_D3 | I | 2 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| M1 | H2 | H1 | VDDANA | GPIO_AD | PD11 | I/O | PTC_Y0 | – | A | TIOA1 | I/O | 3 | PIO, I, PU, ST |
| | | | | | | | | | B | PCK2 | O | 2 | |
| | | | | | | | | | C | UTMI_LS0 | O | 1 | |
| | | | | | | | | | D | GRXDV | I | 2 | |
| | | | | | | | | | E | ISC_D4 | I | 2 | |
| | | | | | | | | | F | ISC_MCK | O | 4 | |
| K8 | K5 | H6 | VDDANA | GPIO_AD | PD12 | I/O | PTC_Y1 | – | A | TIOB1 | I/O | 3 | PIO, I, PU, ST |
| | | | | | | | | | B | FLEXCOM4_IO0 | I/O | 2 | |
| | | | | | | | | | C | UTMI_LS1 | O | 1 | |
| | | | | | | | | | D | GRXER | I | 2 | |
| | | | | | | | | | E | ISC_D5 | I | 2 | |
| | | | | | | | | | F | ISC_D4 | I | 4 | |
| L2 | J5 | H3 | VDDANA | GPIO_AD | PD13 | I/O | PTC_Y2 | – | A | TCLK1 | I | 3 | PIO, I, PU, ST |
| | | | | | | | | | B | FLEXCOM4_IO1 | I/O | 2 | |
| | | | | | | | | | C | UTMI_CDRCPSEL0 | I | 1 | |
| | | | | | | | | | D | GRX0 | I | 2 | |
| | | | | | | | | | E | ISC_D6 | I | 2 | |
| | | | | | | | | | F | ISC_D5 | I | 4 | |
| K4 | K6 | G6 | VDDANA | GPIO_AD | PD14 | I/O | PTC_Y3 | – | A | TCK(4) | I | 1 | A, PU, ST |
| | | | | | | | | | B | FLEXCOM4_IO2 | I/O | 2 | |
| | | | | | | | | | C | UTMI_CDRCPSEL1 | I | 1 | |
| | | | | | | | | | D | GRX1 | I | 2 | |
| | | | | | | | | | E | ISC_D7 | I | 2 | |
| | | | | | | | | | F | ISC_D6 | I | 4 | |
| K7 | K4 | H5 | VDDANA | GPIO_AD | PD15 | I/O | PTC_Y4 | – | A | TDI(4) | I | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | FLEXCOM4_IO3 | O | 2 | |
| | | | | | | | | | C | UTMI_CDRCPDIVEN | I | 1 | |
| | | | | | | | | | D | GTX0 | O | 2 | |
| | | | | | | | | | E | ISC_PCK | I | 2 | |
| | | | | | | | | | F | ISC_D7 | I | 4 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| L1 | K1 | G1 | VDDANA | GPIO_AD | PD16 | I/O | PTC_Y5 | – | A | TDO(4) | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | FLEXCOM4_IO4 | O | 2 | |
| | | | | | | | | | C | UTMI_CDRBISTEN | I | 1 | |
| | | | | | | | | | D | GTX1 | O | 2 | |
| | | | | | | | | | E | ISC_VSYNC | I | 2 | |
| | | | | | | | | | F | ISC_D8 | I | 4 | |
| K2 | K2 | G2 | VDDANA | GPIO_AD | PD17 | I/O | PTC_Y6 | – | A | TMS(4) | I | 1 | A, PU, ST |
| | | | | | | | | | C | UTMI_CDRCPSELDIV | O | 1 | |
| | | | | | | | | | D | GMDC | O | 2 | |
| | | | | | | | | | E | ISC_HSYNC | I | 2 | |
| | | | | | | | | | F | ISC_D9 | I | 4 | |
| J5 | L5 | G3 | VDDANA | GPIO_AD | PD18 | I/O | PTC_Y7 | – | A | NTRST(4) | I | 1 | PIO, I, PU, ST |
| | | | | | | | | | D | GMDIO | I/O | 2 | |
| | | | | | | | | | E | ISC_FIELD | I | 2 | |
| | | | | | | | | | F | ISC_D10 | I | 4 | |
| K6 | L4 | H4 | VDDANA | GPIO_AD | PD19 | I/O | AD0 | – | A | PCK0 | O | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | TWD1 | I/O | 3 | |
| | | | | | | | | | C | URXD2 | I | 3 | |
| | | | | | | | | | E | I2SC0_CK | I/O | 2 | |
| | | | | | | | | | F | ISC_D11 | I | 4 | |
| M2 | M1 | J1 | VDDANA | GPIO_AD | PD20 | I/O | AD1 | – | A | TIOA2 | I/O | 3 | PIO, I, PU, ST |
| | | | | | | | | | B | TWCK1 | I/O | 3 | |
| | | | | | | | | | C | UTXD2 | O | 3 | |
| | | | | | | | | | E | I2SC0_MCK | O | 2 | |
| | | | | | | | | | F | ISC_PCK | I | 4 | |
| N1 | M2 | K1 | VDDANA | GPIO_AD | PD21 | I/O | AD2 | – | A | TIOB2 | I/O | 3 | PIO, I, PU, ST |
| | | | | | | | | | B | TWD0 | I/O | 4 | |
| | | | | | | | | | C | FLEXCOM4_IO0 | I/O | 3 | |
| | | | | | | | | | E | I2SC0_WS | I/O | 2 | |
| | | | | | | | | | F | ISC_VSYNC | I | 4 | |
| L4 | M4 | J3 | VDDANA | GPIO_AD | PD22 | I/O | AD3 | – | A | TCLK2 | I | 3 | PIO, I, PU, ST |
| | | | | | | | | | B | TWCK0 | I/O | 4 | |
| | | | | | | | | | C | FLEXCOM4_IO1 | I/O | 3 | |
| | | | | | | | | | E | I2SC0_DI0 | I | 2 | |
| | | | | | | | | | F | ISC_HSYNC | I | 4 | |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| M3 | P1 | K2 | VDDANA | GPIO_AD | PD23 | I/O | AD4 | – | A | URXD2 | I | 2 | PIO, I, PU, ST |
| | | | | | | | | | C | FLEXCOM4_IO2 | I/O | 3 | |
| | | | | | | | | | E | I2SC0_DO0 | O | 2 | |
| | | | | | | | | | F | ISC_FIELD | I | 4 | |
| L7 | L6 | – | VDDANA | GPIO_AD | PD24 | I/O | AD5 | – | A | UTXD2 | O | 2 | PIO, I, PU, ST |
| | | | | | | | | | C | FLEXCOM4_IO3 | O | 3 | |
| L6 | M5 | – | VDDANA | GPIO_AD | PD25 | I/O | AD6 | – | A | SPI1_SPCK | I/O | 3 | PIO, I, PU, ST |
| | | | | | | | | | C | FLEXCOM4_IO4 | O | 3 | |
| N2 | N1 | – | VDDANA | GPIO_AD | PD26 | I/O | AD7 | – | A | SPI1_MOSI | I/O | 3 | PIO, I, PU, ST |
| | | | | | | | | | C | FLEXCOM2_IO0 | I/O | 2 | |
| L8 | N2 | – | VDDANA | GPIO_AD | PD27 | I/O | AD8 | – | A | SPI1_MISO | I/O | 3 | PIO, I, PU, ST |
| | | | | | | | | | B | TCK | I | 3 | |
| | | | | | | | | | C | FLEXCOM2_IO1 | I/O | 2 | |
| M4 | P2 | – | VDDANA | GPIO_AD | PD28 | I/O | AD9 | – | A | SPI1_NPCS0 | I/O | 3 | PIO, I, PU, ST |
| | | | | | | | | | B | TDI | I | 3 | |
| | | | | | | | | | C | FLEXCOM2_IO2 | I/O | 2 | |
| N3 | R1 | – | VDDANA | GPIO_AD | PD29 | I/O | AD10 | – | A | SPI1_NPCS1 | O | 3 | PIO, I, PU, ST |
| | | | | | | | | | B | TDO | O | 3 | |
| | | | | | | | | | C | FLEXCOM2_IO3 | O | 2 | |
| | | | | | | | | | D | TIOA3 | I/O | 3 | |
| | | | | | | | | | E | TWD0 | I/O | 3 | |
| L9 | N4 | – | VDDANA | GPIO_AD | PD30 | I/O | AD11 | – | A | SPI1_NPCS2 | O | 3 | PIO, I, PU, ST |
| | | | | | | | | | B | TMS | I | 3 | |
| | | | | | | | | | C | FLEXCOM2_IO4 | O | 2 | |
| | | | | | | | | | D | TIOB3 | I/O | 3 | |
| | | | | | | | | | E | TWCK0 | I/O | 3 | |
| M7 | T1 | – | VDDANA | GPIO | PD31 | I/O | – | – | A | ADTRG | I | 1 | PIO, I, PU, ST |
| | | | | | | | | | B | NTRST | I | 3 | |
| | | | | | | | | | C | IRQ | I | 4 | |
| | | | | | | | | | D | TCLK3 | I | 3 | |
| | | | | | | | | | E | PCK0 | O | 2 | |
| L5 | L1 | K3 | VDDANA | power | VDDANA | I | – | – | – | – | – | – | – |
| K5 | L2 | K4 | GNDANA | ground | GNDANA | I | – | – | – | – | – | – | – |
| M6 | P5 | L2 | VDDANA | – | ADVREF | I | – | – | – | – | – | – | – |
| K3 | J1 | H2 | VDDANA | power | VDDANA | I | – | – | – | – | – | – | – |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| L3 | J2 | J2 | GNDANA | ground | GNDANA | I | – | – | – | – | – | – | – |
| H16, D16 | J17, D12 | H12, C12 | VDDIODDR | DDR | DDR_VREF | I | – | – | – | – | – | – | – |
| B12 | B12 | B7 | VDDIODDR | DDR | DDR_D0 | I/O | – | – | – | – | – | – | – |
| A12 | B13 | A7 | VDDIODDR | DDR | DDR_D1 | I/O | – | – | – | – | – | – | – |
| C12 | D13 | C8 | VDDIODDR | DDR | DDR_D2 | I/O | – | – | – | – | – | – | – |
| A13 | A13 | B9 | VDDIODDR | DDR | DDR_D3 | I/O | – | – | – | – | – | – | – |
| A14 | A15 | A9 | VDDIODDR | DDR | DDR_D4 | I/O | – | – | – | – | – | – | – |
| C13 | D14 | C9 | VDDIODDR | DDR | DDR_D5 | I/O | – | – | – | – | – | – | – |
| A15 | B15 | A10 | VDDIODDR | DDR | DDR_D6 | I/O | – | – | – | – | – | – | – |
| B15 | B16 | B10 | VDDIODDR | DDR | DDR_D7 | I/O | – | – | – | – | – | – | – |
| G17 | G18 | H13 | VDDIODDR | DDR | DDR_D8 | I/O | – | – | – | – | – | – | – |
| G16 | K17 | H14 | VDDIODDR | DDR | DDR_D9 | I/O | – | – | – | – | – | – | – |
| H17 | J13 | J13 | VDDIODDR | DDR | DDR_D10 | I/O | – | – | – | – | – | – | – |
| K17 | H15 | J14 | VDDIODDR | DDR | DDR_D11 | I/O | – | – | – | – | – | – | – |
| K16 | J15 | L13 | VDDIODDR | DDR | DDR_D12 | I/O | – | – | – | – | – | – | – |
| J13 | J14 | L14 | VDDIODDR | DDR | DDR_D13 | I/O | – | – | – | – | – | – | – |
| K14 | K13 | J12 | VDDIODDR | DDR | DDR_D14 | I/O | – | – | – | – | – | – | – |
| K15 | K18 | K12 | VDDIODDR | DDR | DDR_D15 | I/O | – | – | – | – | – | – | – |
| B8 | A8 | – | VDDIODDR | DDR | DDR_D16 | I/O | – | – | – | – | – | – | – |
| B9 | B9 | – | VDDIODDR | DDR | DDR_D17 | I/O | – | – | – | – | – | – | – |
| C9 | D9 | – | VDDIODDR | DDR | DDR_D18 | I/O | – | – | – | – | – | – | – |
| A9 | A9 | – | VDDIODDR | DDR | DDR_D19 | I/O | – | – | – | – | – | – | – |
| A10 | B11 | – | VDDIODDR | DDR | DDR_D20 | I/O | – | – | – | – | – | – | – |
| D10 | D10 | – | VDDIODDR | DDR | DDR_D21 | I/O | – | – | – | – | – | – | – |
| B11 | A11 | – | VDDIODDR | DDR | DDR_D22 | I/O | – | – | – | – | – | – | – |
| A11 | A12 | – | VDDIODDR | DDR | DDR_D23 | I/O | – | – | – | – | – | – | – |
| J12 | L18 | – | VDDIODDR | DDR | DDR_D24 | I/O | – | – | – | – | – | – | – |
| H10 | K15 | – | VDDIODDR | DDR | DDR_D25 | I/O | – | – | – | – | – | – | – |
| J11 | K14 | – | VDDIODDR | DDR | DDR_D26 | I/O | – | – | – | – | – | – | – |
| K11 | M18 | – | VDDIODDR | DDR | DDR_D27 | I/O | – | – | – | – | – | – | – |
| L13 | N17 | – | VDDIODDR | DDR | DDR_D28 | I/O | – | – | – | – | – | – | – |
| L11 | M14 | – | VDDIODDR | DDR | DDR_D29 | I/O | – | – | – | – | – | – | – |
| L12 | M15 | – | VDDIODDR | DDR | DDR_D30 | I/O | – | – | – | – | – | – | – |
| M17 | N18 | – | VDDIODDR | DDR | DDR_D31 | I/O | – | – | – | – | – | – | – |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| F12 | D17 | E11 | VDDIODDR | DDR | DDR_A0 | O | – | – | – | – | – | – | – |
| C17 | A17 | C11 | VDDIODDR | DDR | DDR_A1 | O | – | – | – | – | – | – | – |
| B17 | A18 | B12 | VDDIODDR | DDR | DDR_A2 | O | – | – | – | – | – | – | – |
| B16 | F15 | A12 | VDDIODDR | DDR | DDR_A3 | O | – | – | – | – | – | – | – |
| C16 | G12 | D11 | VDDIODDR | DDR | DDR_A4 | O | – | – | – | – | – | – | – |
| G14 | H12 | D14 | VDDIODDR | DDR | DDR_A5 | O | – | – | – | – | – | – | – |
| F14 | F13 | B14 | VDDIODDR | DDR | DDR_A6 | O | – | – | – | – | – | – | – |
| F11 | H10 | D9 | VDDIODDR | DDR | DDR_A7 | O | – | – | – | – | – | – | – |
| C14 | A16 | C10 | VDDIODDR | DDR | DDR_A8 | O | – | – | – | – | – | – | – |
| D13 | E12 | D10 | VDDIODDR | DDR | DDR_A9 | O | – | – | – | – | – | – | – |
| C15 | H11 | F9 | VDDIODDR | DDR | DDR_A10 | O | – | – | – | – | – | – | – |
| A16 | J10 | A11 | VDDIODDR | DDR | DDR_A11 | O | – | – | – | – | – | – | – |
| A17 | D15 | B11 | VDDIODDR | DDR | DDR_A12 | O | – | – | – | – | – | – | – |
| G11 | J11 | E13 | VDDIODDR | DDR | DDR_A13 | O | – | – | – | – | – | – | – |
| E17 | C18 | A13 | VDDIODDR | DDR | DDR_CLK | O | – | – | – | – | – | – | – |
| D17 | C17 | B13 | VDDIODDR | DDR | DDR_CLKN | O | – | – | – | – | – | – | – |
| F16 | F18 | E14 | VDDIODDR | DDR | DDR_CKE | O | – | – | – | – | – | – | – |
| E16 | F17 | D13 | VDDIODDR | DDR | DDR_RESETN | O | – | – | – | – | – | – | – |
| G13 | J12 | F11 | VDDIODDR | DDR | DDR_CS | O | – | – | – | – | – | – | – |
| F15 | D18 | A14 | VDDIODDR | DDR | DDR_WE | O | – | – | – | – | – | – | – |
| F13 | E18 | C14 | VDDIODDR | DDR | DDR_RAS | O | – | – | – | – | – | – | – |
| G12 | E17 | C13 | VDDIODDR | DDR | DDR_CAS | O | – | – | – | – | – | – | – |
| C11 | D11 | D8 | VDDIODDR | DDR | DDR_DQM0 | O | – | – | – | – | – | – | – |
| G15 | H14 | G14 | VDDIODDR | DDR | DDR_DQM1 | O | – | – | – | – | – | – | – |
| C8 | B8 | – | VDDIODDR | DDR | DDR_DQM2 | O | – | – | – | – | – | – | – |
| H11 | L13 | – | VDDIODDR | DDR | DDR_DQM3 | O | – | – | – | – | – | – | – |
| B13 | A14 | B8 | VDDIODDR | DDR | DDR_DQS0 | O | – | – | – | – | – | – | – |
| J17 | H18 | K14 | VDDIODDR | DDR | DDR_DQS1 | O | – | – | – | – | – | – | – |
| C10 | A10 | – | VDDIODDR | DDR | DDR_DQS2 | O | – | – | – | – | – | – | – |
| L17 | M17 | – | VDDIODDR | DDR | DDR_DQS3 | O | – | – | – | – | – | – | – |
| B14 | B14 | A8 | VDDIODDR | DDR | DDR_DQSN0 | O | – | – | – | – | – | – | – |
| J16 | J18 | K13 | VDDIODDR | DDR | DDR_DQSN1 | O | – | – | – | – | – | – | – |
| B10 | B10 | – | VDDIODDR | DDR | DDR_DQSN2 | O | – | – | – | – | – | – | – |
| L16 | L17 | – | VDDIODDR | DDR | DDR_DQSN3 | O | – | – | – | – | – | – | – |
| H12 | H13 | F13 | VDDIODDR | DDR | DDR_BA0 | O | – | – | – | – | – | – | – |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| H13 | K12 | G13 | VDDIODDR | DDR | DDR_BA1 | O | – | – | – | – | – | – | – |
| F17 | H17 | F14 | VDDIODDR | DDR | DDR_BA2 | O | – | – | – | – | – | – | – |
| E13 | G17 | F10 | VDDIODDR | DDR | DDR_CAL | I | – | – | – | – | – | – | – |
| L15, J15, H15, E15, D15, D12, D11 | B17, E11, E14, F10, G11, G15, L14 | C6, E10, E12, G10, G12, H11, J10 | VDDIODDR | power | VDDIODDR | I | – | – | – | – | – | – | – |
| L14, J14, H14, E14, D14, E12, E11 | B18, E10, E15, F11, G10, G14, L15 | C7, D12, E9, F12, G11, H10, J11 | GNDIODDR | ground | GNDIODDR | I | – | – | – | – | – | – | – |
| H3, N5, N9, K13, D9, D7 | H8, J6, J9, K8, L8 | E8, G8, H8, H9, J5 | VDDCORE | power | VDDCORE | I | – | – | – | – | – | – | – |
| H4, M5, M9, K12, E9, E7 | H9, J7, J8, K7, L7 | F8, G7, G9, H7, J4 | GNDCORE | ground | GNDCORE | I | – | – | – | – | – | – | – |
| E6, F7 | B1, D5 | D7, F4 | VDDIOP0 | power | VDDIOP0 | I | – | – | – | – | – | – | – |
| F6, G7 | B2, D4 | E4, E7 | GNDIOP0 | ground | GNDIOP0 | I | – | – | – | – | – | – | – |
| R14, N13 | T18, V16 | K8, L11 | VDDIOP1 | power | VDDIOP1 | I | – | – | – | – | – | – | – |
| M13, P14 | T17, V15 | K9, L12 | GNDIOP1 | ground | GNDIOP1 | I | – | – | – | – | – | – | – |
| F10 | D8 | – | VDDIOP2 | power | VDDIOP2 | I | – | – | – | – | – | – | – |
| F9 | E8 | – | GNDIOP2 | ground | GNDIOP2 | I | – | – | – | – | – | – | – |
| P11 | R11 | – | VDDSDMMC | power | VDDSDMMC | I | – | – | – | – | – | – | – |
| R11 | R12 | – | GNDSDMMC | ground | GNDSDMMC | I | – | – | – | – | – | – | – |
| F4 | – | – | VDDISC | power | VDDISC | I | – | – | – | – | – | – | – |
| G4 | – | – | GNDISC | ground | GNDISC | I | – | – | – | – | – | – | – |
| M12 | R17 | K11 | VDDFUSE | power | VDDFUSE | I | – | – | – | – | – | – | – |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| U4 | V5 | P3 | VDDPLLA | power | VDDPLLA | I | – | – | – | – | – | – | – |
| U5 | U6 | P4 | GNDPLLA | ground | GNDPLLA | I | – | – | – | – | – | – | – |
| T3 | M7 | K6 | VDDAUDIOPLL | power | VDDAUDIOPLL | I | – | – | – | – | – | – | – |
| T5 | P7 | L6 | GNDDPLL | ground | GNDDPLL | I | – | – | – | – | – | – | – |
| T4 | N6 | J6 | GNDAUDIOPLL | ground | GNDAUDIOPLL | I | – | – | – | – | – | – | – |
| U3 | M8 | J7 | VDDAUDIOPLL | – | CLK_AUDIO | O | – | – | – | – | – | – | – |
| U7 | V7 | P5 | VDDOSC | – | XIN | I | – | – | – | – | – | – | – |
| U6 | V6 | P6 | VDDOSC | – | XOUT | O | – | – | – | – | – | – | – |
| T7 | R8 | N5 | VDDOSC | power | VDDOSC | I | – | – | – | – | – | – | – |
| T6 | U5 | N6 | GNDOSC | ground | GNDOSC | I | – | – | – | – | – | – | – |
| P8 | N8 | K7 | VDDUTMII | power | VDDUTMII | I | – | – | – | – | – | – | – |
| R9 | P9 | – | VDDHSIC | power | VDDHSIC | I | – | – | – | – | – | – | – |
| P9 | N9 | L8 | GNDUTMII | ground | GNDUTMII | I | – | – | – | – | – | – | – |
| T8 | U8 | N7 | VDDUTMII | – | HHSDPA | I/O | – | – | – | – | – | – | – |
| R8 | V8 | P7 | VDDUTMII | – | HHSDMA | I/O | – | – | – | – | – | – | – |
| U8 | U9 | N8 | VDDUTMII | – | HHSDPB | I/O | – | – | – | – | – | – | – |
| U9 | V9 | P8 | VDDUTMII | – | HHSDMB | I/O | – | – | – | – | – | – | – |
| T9 | U10 | – | VDDHSIC | – | HHSDPDATC | I/O | – | – | – | – | – | – | – |
| U10 | V10 | – | VDDHSIC | – | HHSDMSTRC | I/O | – | – | – | – | – | – | – |
| P7 | P8 | M7 | VDDUTMIC | power | VDDUTMIC | I | – | – | – | – | – | – | – |
| R7 | U7 | M8 | GNDUTMIC | ground | GNDUTMIC | I | – | – | – | – | – | – | – |
| T10 | N10 | – | VDDSDMMC | – | SDCAL | I | – | – | – | – | – | – | – |
| R6 | R7 | L7 | VDDUTMIC | – | VBG | I | – | – | – | – | – | – | – |
| P3 | P4 | M2 | VDDBU | – | TST | I | – | – | – | – | – | – | – |
| U2 | V1 | N3 | VDDBU | – | NRST(3) | I | – | – | – | – | – | – | – |
| T2 | V2 | L4 | VDDBU | – | JTAGSEL | I | – | – | – | – | – | – | – |
| P4 | R5 | P1 | VDDBU | – | WKUP | I | – | – | – | – | – | – | – |
| N4 | U2 | – | VDDBU | – | RXD | I | – | – | – | – | – | – | – |
| R1 | U1 | N1 | VDDBU | – | SHDN | O | – | – | – | – | – | – | – |
| R3 | R6 | K5 | VDDBU | – | PIOBU0 | I/O | – | – | – | – | – | – | – |
| N8 | R4 | L3 | VDDBU | – | PIOBU1 | I/O | – | – | – | – | – | – | – |
| R2 | – | M3 | VDDBU | – | PIOBU2 | I/O | – | – | – | – | – | – | – |
| R5 | – | N4 | VDDBU | – | PIOBU3 | I/O | – | – | – | – | – | – | – |
| R4 | – | L5 | VDDBU | – | PIOBU4 | I/O | – | – | – | – | – | – | – |
| P5 | – | M6 | VDDBU | – | PIOBU5 | I/O | – | – | – | – | – | – | – |

© 2018 Microchip Technology Inc.

.....continued

| 289-pin BGA | 256-pin BGA | 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| P6 | – | – | VDDBU | – | PIOBU6 | I/O | – | – | – | – | – | – | – |
| M8 | – | – | VDDBU | – | PIOBU7 | I/O | – | – | – | – | – | – | – |
| N7 | U3 | M4 | VDDBU | power | VDDBU | I | – | – | – | – | – | – | – |
| N6 | U4 | M5 | GNDBU | ground | GNDBU | I | – | – | – | – | – | – | – |
| P1 | T2 | M1 | VDDBU | – | XIN32 | I | – | – | – | – | – | – | – |
| P2 | R2 | L1 | VDDBU | – | XOUT32 | O | – | – | – | – | – | – | – |
| T1 | V3 | N2 | VDDBU | – | COMPP | I | – | – | – | – | – | – | – |
| U1 | V4 | P2 | VDDBU | – | COMPN | I | – | – | – | – | – | – | – |

© 2018 Microchip Technology Inc.

**Note:**

1. Signal = ‘PIO’ if GPIO; Dir = Direction; PU = Pull-up; PD = Pull-down; HiZ = High impedance; ST = Schmitt Trigger
2. The GPIO reset state is not guaranteed during the power-up phase. During this phase, the GPIOs are in Input Pull-Up mode and they take their reset value only after VDDCORE POR reset has been released. If a GPIO must be at level zero at power-up, it is recommended to connect an external pull-down to ensure this state.
3. For NRST usage, refer to the section Reset and Test.
4. JTAG boundary scan is available only on JTAG IO Set 1.

The SAMA5D23 is not pin-to-pin compatible with SAMA5D21/SAMA5D22. The table below provides the differences in pinout.

**Table 6-3. Pin Description (SAMA5D23 pins different from those in table Pin Description (all packages))**

| 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| N4 | GNDBU | ground | GNDBU | I | – | – | – | – | – | – | – |
| M6 | GNDDPLL | ground | GNDDPLL | I | – | – | – | – | – | – | – |
| M3 | JTAGSEL | – | JTAGSEL | I | – | – | – | – | – | – | – |
| K11 | VDDIOP1 | GPIO | PA31 | I/O | – | – | B | NCS3 | O | 1 | PIO, I, PU, ST |
| | | | | | | | C | SPI0_MISO | I/O | 2 | |
| | | | | | | | D | PWML0 | O | 1 | |
| | | | | | | | F | CLASSD_L3 | O | 1 | |
| D6 | VDDIOP0 | GPIO | PB0 | I/O | – | – | B | A21/NANDALE | O | 1 | PIO, I, PU, ST |
| | | | | | | | C | SPI0_MOSI | I/O | 2 | |
| | | | | | | | D | PWMH1 | O | 1 | |
| A6 | VDDIOP0 | GPIO | PB2 | I/O | – | – | B | NRD/NANDOE | O | 1 | PIO, I, PU, ST |
| | | | | | | | D | PWMFI0 | I | 1 | |
| | | | | | | | F | CLASSD_R1 | O | 1 | |
| B6 | VDDIOP0 | GPIO | PB3 | I/O | – | – | A | URXD4 | I | 1 | PIO, I, PU, ST |
| | | | | | | | B | D8 | I/O | 1 | |
| | | | | | | | C | IRQ | I | 3 | |
| | | | | | | | D | PWMEXTRG1 | I | 1 | |
| | | | | | | | F | CLASSD_R2 | O | 1 | |
| B5 | VDDIOP0 | GPIO_QSPI | PB5 | I/O | – | – | A | TCLK2 | I | 1 | PIO, I, PU, ST |
| | | | | | | | B | D10 | I/O | 1 | |
| | | | | | | | C | PWMH2 | O | 1 | |
| | | | | | | | D | QSPI1_SCK | O | 2 | |
| | | | | | | | F | GTSUCOMP | O | 3 | |
| M12 | VDDIOP1 | GPIO | PC0 | I/O | – | – | A | LCDDAT21 | O | 1 | PIO, I, PU, ST |
| | | | | | | | B | A23 | O | 1 | |
| | | | | | | | C | FLEXCOM0_IO4 | O | 1 | |
| | | | | | | | D | TWCK0 | I/O | 1 | |
| | | | | | | | F | ISC_D6 | I | 3 | |
| M13 | VDDIOP1 | GPIO | PC1 | I/O | – | – | A | LCDDAT22 | O | 1 | PIO, I, PU, ST |
| | | | | | | | B | A24 | O | 1 | |
| | | | | | | | C | CANTX0 | O | 1 | |
| | | | | | | | D | SPI1_SPCK | I/O | 1 | |
| | | | | | | | E | I2SC0_CK | I/O | 1 | |
| | | | | | | | F | ISC_D7 | I | 3 | |
| L4 | VDDBU | – | PIOBU1 | I/O | – | – | – | – | – | – | – |

© 2018 Microchip Technology Inc.

.....continued

| 196-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| L3 | VDDBU | – | PIOBU2 | I/O | – | – | – | – | – | – | – |
| M5 | VDDBU | – | PIOBU3 | I/O | – | – | – | – | – | – | – |
| L6 | VDDBU | – | PIOBU5 | I/O | – | – | – | – | – | – | – |
| P13 | VDDFUSE | power | VDDFUSE | I | – | – | – | – | – | – | – |

© 2018 Microchip Technology Inc.

**Note:**

1. Signal = ‘PIO’ if GPIO; Dir = Direction; PU = Pull-up; PD = Pull-down; HiZ = High impedance; ST = Schmitt Trigger.
2. The GPIO reset state is not guaranteed during the power-up phase. During this phase, the GPIOs are in Input Pull-Up mode and they take their reset value only after VDDCORE POR reset has been released. If a GPIO must be at level zero at power-up, it is recommended to connect an external pull-down to ensure this state.

The SAMA5D28B/C are not pin-to-pin compatible with SAMA5D28A, SAMA5D26A/B/C and SAMA5D27A/B/C. The table below provides the differences in pinout.

**Table 6-4. Pin Description (SAMA5D28B/C pins different from those in the table Pin Description (all packages))**

| 289-pin BGA | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral | | | | Reset State (Signal, Dir, PU, PD, HiZ, ST)(1)(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Signal | Dir | Signal | Dir | Func | Signal | Dir | IO Set | |
| P4 | VDDCORE | power | VDDCORE | I | – | – | – | – | – | – | – |
| N5 | GNDCORE | ground | GNDCORE | I | – | – | – | – | – | – | – |
| R2 | VDDBU | – | WKUP | I | – | – | – | – | – | – | – |
| N6 | VDDBU | – | PIOBU0 | I/O | – | – | – | – | – | – | – |
| M8 | VDDBU | – | PIOBU2 | I/O | – | – | – | – | – | – | – |
| P6 | VDDBU | – | PIOBU3 | I/O | – | – | – | – | – | – | – |
| P5 | VDDBU | – | PIOBU4 | I/O | – | – | – | – | – | – | – |
| R5 | VDDBU | – | PIOBU5 | I/O | – | – | – | – | – | – | – |
| N7 | VDDBU | – | PIOBU6 | I/O | – | – | – | – | – | – | – |
| M5 | VDDBU | – | PIOBU7 | I/O | – | – | – | – | – | – | – |
| R3 | VDDBU | power | VDDBU | I | – | – | – | – | – | – | – |
| R4 | GNDBU | ground | GNDBU | I | – | – | – | – | – | – | – |

© 2018 Microchip Technology Inc.

**Note:**

1. Signal = ‘PIO’ if GPIO; Dir = Direction; PU = Pull-up; PD = Pull-down; HiZ = High impedance; ST = Schmitt Trigger.
2. The GPIO reset state is not guaranteed during the power-up phase. During this phase, the GPIOs are in Input Pull-Up mode and they take their reset value only after VDDCORE POR reset has been released. If a GPIO must be at level zero at power-up, it is recommended to connect an external pull-down to ensure this state.

# 7. Power Considerations

## 7.1 Power Supplies

**Table 7-1. SAMA5D2 Power Supplies**

| Name | Voltage Range, Nominal | Associated Ground | Powers |
|---|---|---|---|
| VDDCORE | 1.10V – 1.32V, 1.20V | GNDCORE | Core, including the processor, the embedded memories and the peripherals |
| VDDPLLA | 1.10V – 1.32V, 1.20V | GNDPLLA | PLLA Cell |
| VDDUTMIC | 1.10V – 1.32V, 1.20V | GNDUTMII | USB device and host UTMI+ core |
| VDDHSIC | 1.10V – 1.30V, 1.20V | GNDUTMII | USB High-Speed Inter-Chip |
| VDDIODDR | 1.70V – 1.90V, 1.80V<br>1.14V – 1.30V, 1.20V<br>1.29V – 1.45V, 1.35V<br>1.43V – 1.57V, 1.50V | GNDIODDR | LPDDR1 / DDR2 Interface I/O lines<br>LPDDR2 / LPDDR3 Interface I/O lines<br>DDR3L Interface I/O lines<br>DDR3 Interface I/O lines |
| VDDIOP0 | 1.65V – 3.60V | GNDIOP0 | Peripheral I/O lines |
| VDDIOP1 | 1.65V – 3.60V | GNDIOP1 | Peripheral I/O lines |
| VDDIOP2 | 1.65V – 3.60V | GNDIOP2 | Peripheral I/O lines |
| VDDISC | 1.65V – 3.60V | GNDISC | Image Sensor I/O lines |
| VDDSDMMC | 1.65V – 3.60V | GNDSDMMC | SDMMC I/O lines |
| VDDUTMII | 3.00V – 3.60V, 3.30V | GNDUTMII | USB device and host UTMI+ interface |
| VDDOSC | 1.65V – 3.60V | GNDOSC | Main Oscillator cell and PLL UTMI. If PLL UTMI or USB is used, the range is restricted to 3.00V–3.60V |
| VDDAUDIOPLL | 3.00V – 3.60V, 3.30V | GNDAUDIOPLL<br>GNDDPLL | Audio PLL |
| VDDANA | 1.65V – 3.60V, 3.30V | GNDANA | VDD Analog |
| VDDFUSE | 2.25V – 2.75V, 2.50V | GNDFUSE | Fuse box for programming. It can be tied to ground with a 100 Ω resistor for fuse reading only. It must be powered for fuse programming and to switch to Secure mode. |
| VDDBU | 1.65V – 3.60V | GNDBU | Slow Clock Oscillator, the internal 64-kHz RC Oscillator and a part of the System Controller |

## 7.2 Power-up Considerations

At power-up, from a supply sequencing perspective, the SAMA5D2 power supply inputs are categorized into two groups:

- Group 1 (core group) contains VDDCORE, VDDUTMIC, VDDHSIC and VDDPLLA.
- Group 2 (periphery group) contains all other power supply inputs except VDDFUSE.
  The figure below shows the recommended power-up sequence. Note that:

- VDDBU, when supplied from a battery, is an always-on supply input and is therefore not part of the power supply sequencing. When no backup battery is present in the application, VDDBU is part of Group 2.
- VDDFUSE is the only power supply that may be left unpowered during operation. This is possible if and only if the application does not access the Customer Fuse Matrix in Write mode. It is good practice to turn on VDDFUSE only when the Customer Fuse Matrix is accessed in Write mode, and to turn off VDDFUSE otherwise.
- VDDIODDR may be nominally supplied at 1.2V when the SAMA5D2 device is equipped with an LPDDR2 or LPDDR3 memory. In this case, VDDIODDR can be considered as part of Group 1.

**Figure 7-1. Recommended Power-up Sequence**

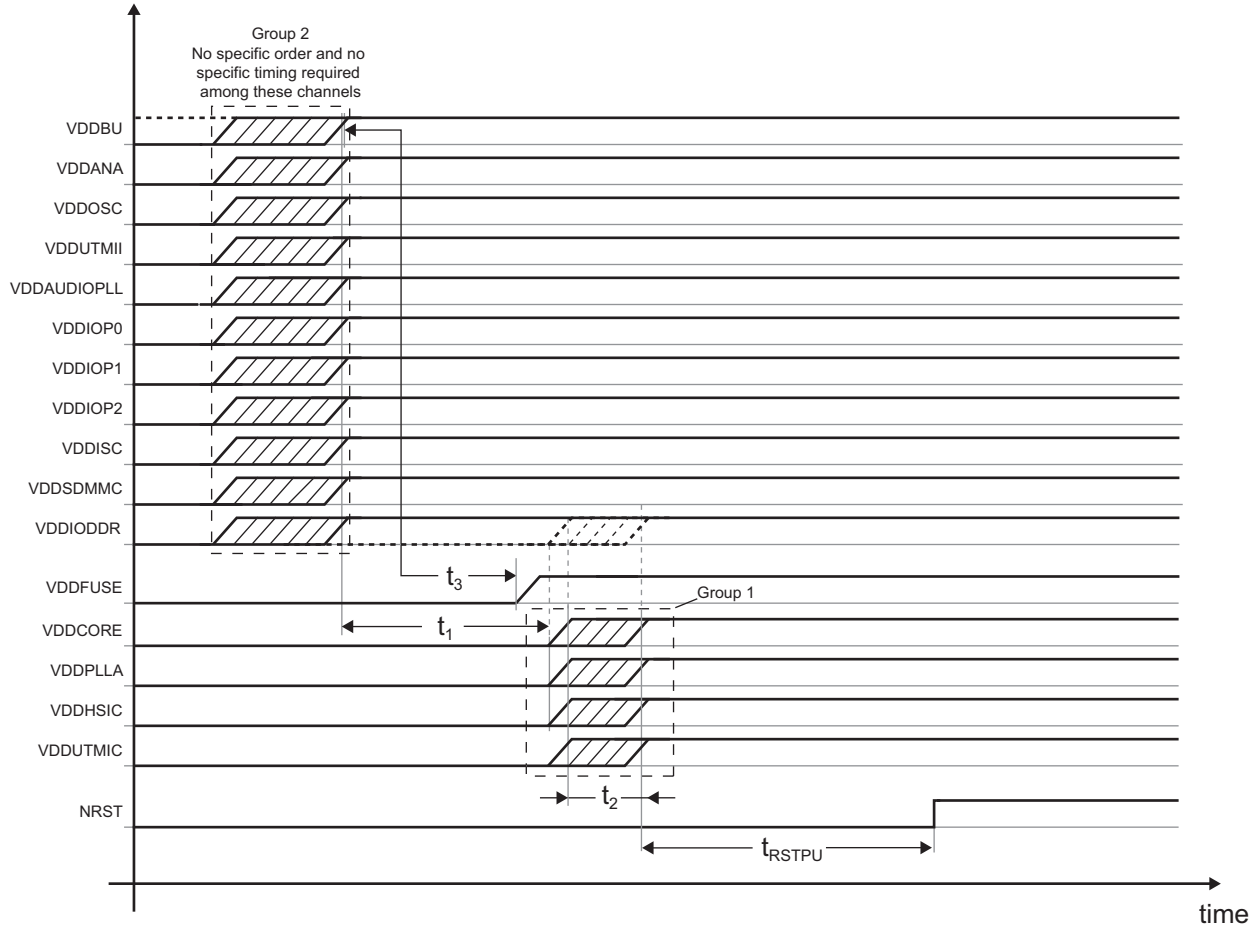


**Table 7-2. Power-up Timing Specification**

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $t_1$ | Group 2 to Group 1 delay | Delay from the last Group 2 established(1) supply to the first Group1 supply turn-on | 0 | – | ms |
| $t_2$ | Group 1 delay(2) | Delay from the first group 1 established supply to the last Group 1 established supply | – | 1 | |
| $t_3$ | VDDFUSE to VDDBU delay | Delay from VDDBU established to VDDFUSE turn-on | 1 | – | |
| $t_{RSTPU}$ | Reset delay at power-up | From the last established supply to NRST high | 1 | – | |

**Note:**

1. An “established” supply refers to a power supply established at 90% of its final value.
2. Also applies to VDDIODDR when considered as part of Group 1.

## 7.3 Power-down Considerations

The figure below shows the SAMA5D2 power-down sequence that starts by asserting the NRST line to 0. Once NRST is asserted, the supply inputs can be immediately shut down without any specific timing or order. VDDBU may not be shut down if the application uses a backup battery on this supply input. In applications where VDDFUSE is powered, it is mandatory to shut down VDDFUSE prior to removing any other supply. VDDFUSE can be removed before or after asserting the NRST signal.

**Figure 7-2. Recommended Power-down Sequence**

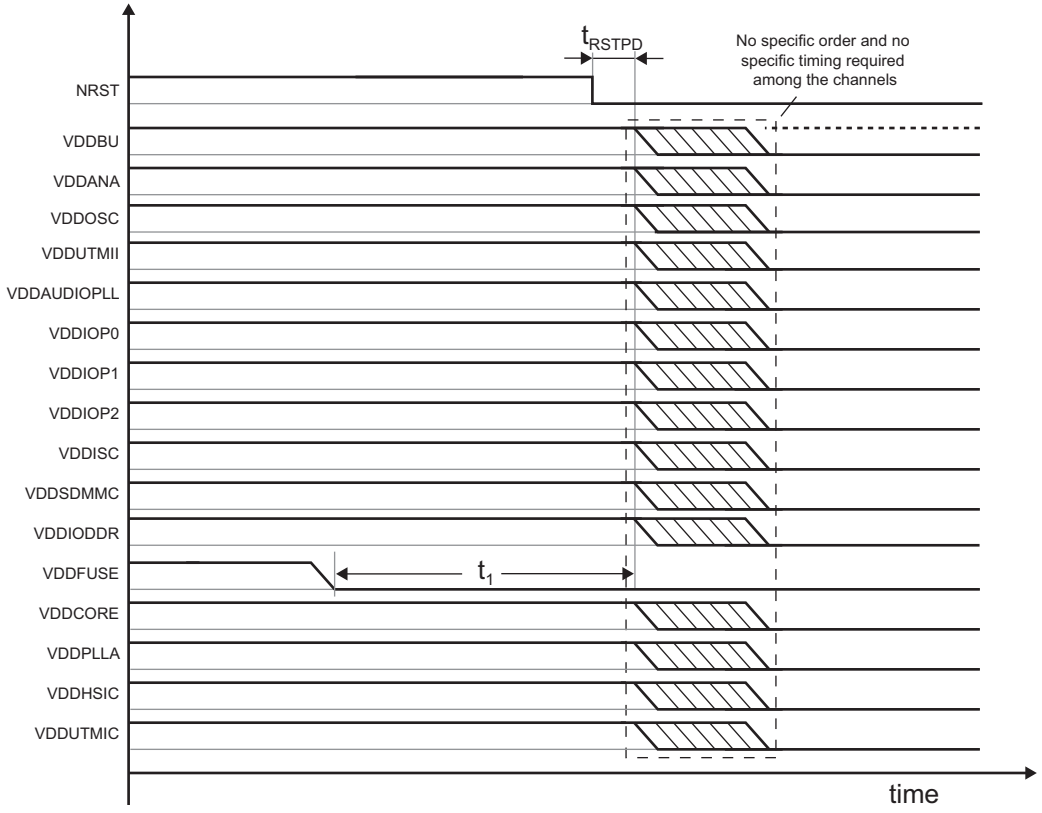


**Table 7-3. Power-down Timing Specification**

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $t_{RSTPD}$ | Reset delay at power-down | From NRST low to the first supply turn-off | 0 | – | ms |
| $t_1$ | VDDFUSE delay at shut-down | From VDDFUSE < 1V to the first supply turn-off | 0 | – | |

## 7.4 Power Supply Sequencing at Backup Mode Entry and Exit

### 7.4.1 VDDBU Power Architecture

The backup power switch aims at optimizing the power consumption on VDDBU source by switching the supply of the backup digital part (BUREG memories + 64-kHz RC oscillator) to VDDANA.

When enabled, the backup power source can be automatically switched to VDDANA, which reduces power consumption on VDDBU. Then, VDDBU powers the pads, VDDBU POR, 32-kHz crystal and, on secure products SAMA5D23 and SAMA5D28, the temperature sensor and the backup supply monitor.

The power source (VDDANA or VDDBU) can be selected manually or can be set to work automatically by programming an SFRBU register (refer to SFRBU_PSWBUCTRL in the section Special Function Registers Backup (SFRBU)).

### 7.4.2 Backup Mode Entry

The figure below shows the recommended power down sequence to place the SAMA5D2 either in Backup mode or in Backup mode with its DDR in self-refresh. The SHDN signal, output of Shutdown Controller (SHDWC), signals the shutdown request to the power supply. This output is supplied by VDDBU that is present in Backup mode. Placing the external DDR memory in self-refresh while in Backup mode, requires to maintain also VDDIODDR. One possible way to signal this additional need to the power supply is to position one of the general purpose I/Os supplied by VDDBU (PIOBUx) in a predefined state.

**Figure 7-3. Recommended Backup Mode Entry**

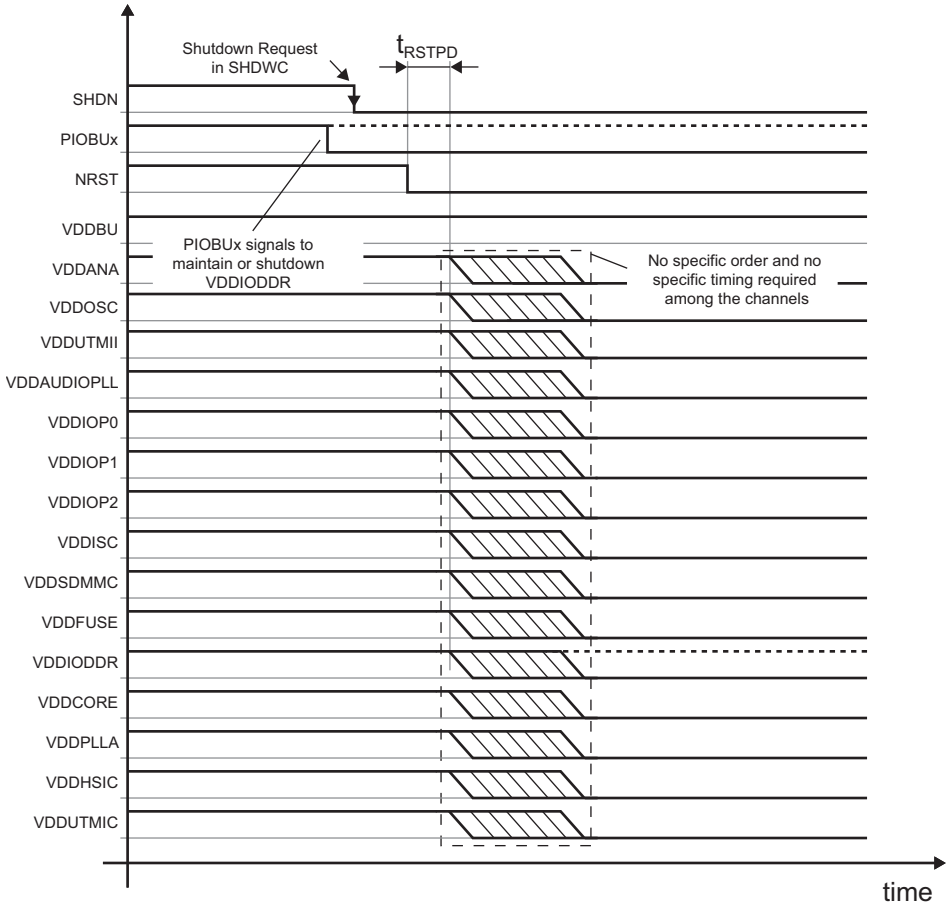



**Table 7-4. Powerdown Timing Specification**

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $t_{RSTPD}$ | Reset delay at powerdown | From NRST low to the first supply turn-off | 0 | – | ms |

### 7.4.3 Backup Mode Exit (Wake-up)

The figure below shows the recommended power-up sequence to wake up SAMA5D2 from Backup mode. Upon a wake-up event, the Shutdown Controller toggles its SHDN output back to VDDBU to request the power supply to restart. Except for VDDIODDR which may already be present if the external DDR memory was placed in Self-refresh mode, this power-up sequence is the same one as presented in the figure “Recommended Power-up Sequence”. In particular, the definitions of Group 1 and Group 2 are the same.

**Figure 7-4. Recommended Power Supply Sequencing at Wake-up**

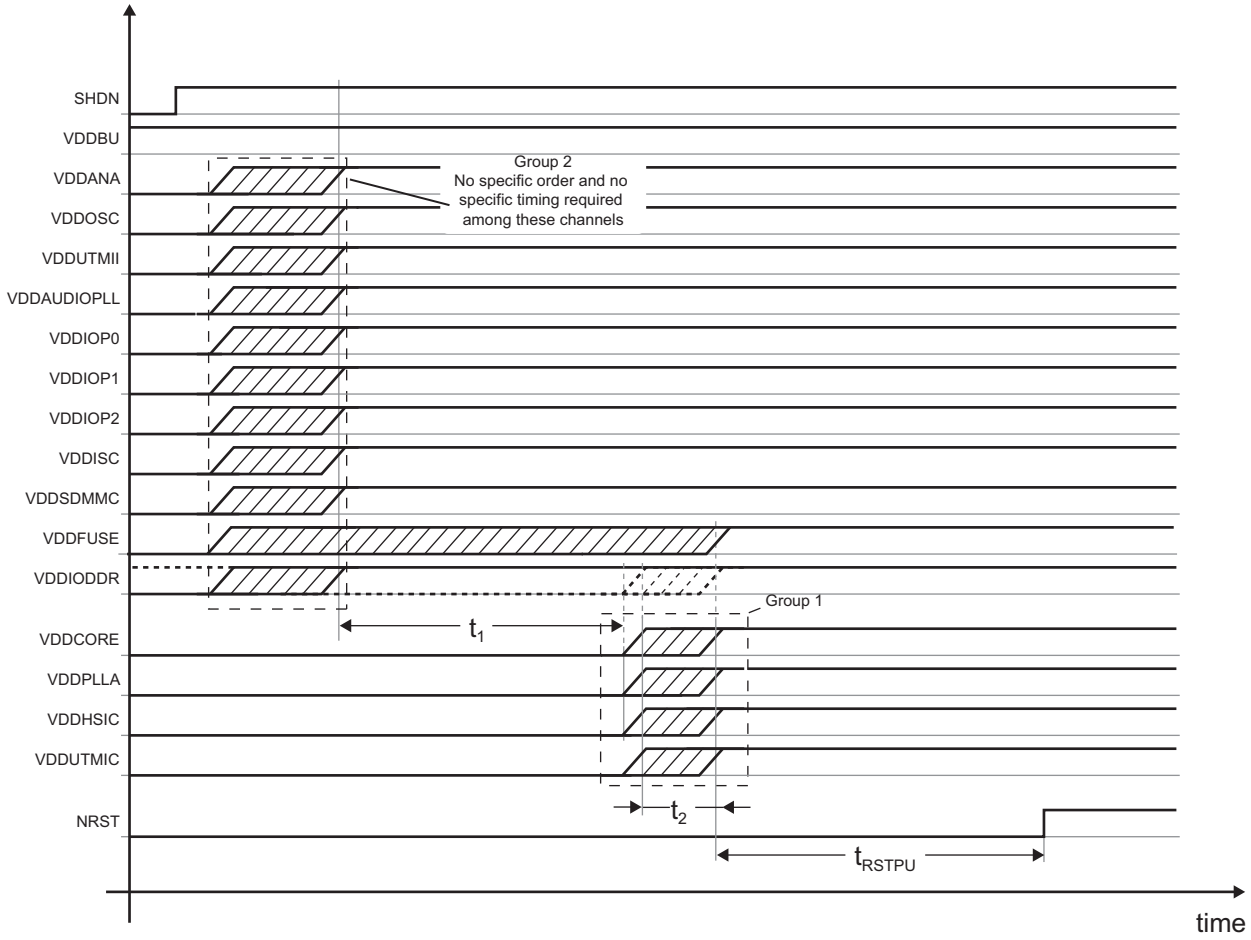



**Table 7-5. Power-up Timing Specification**

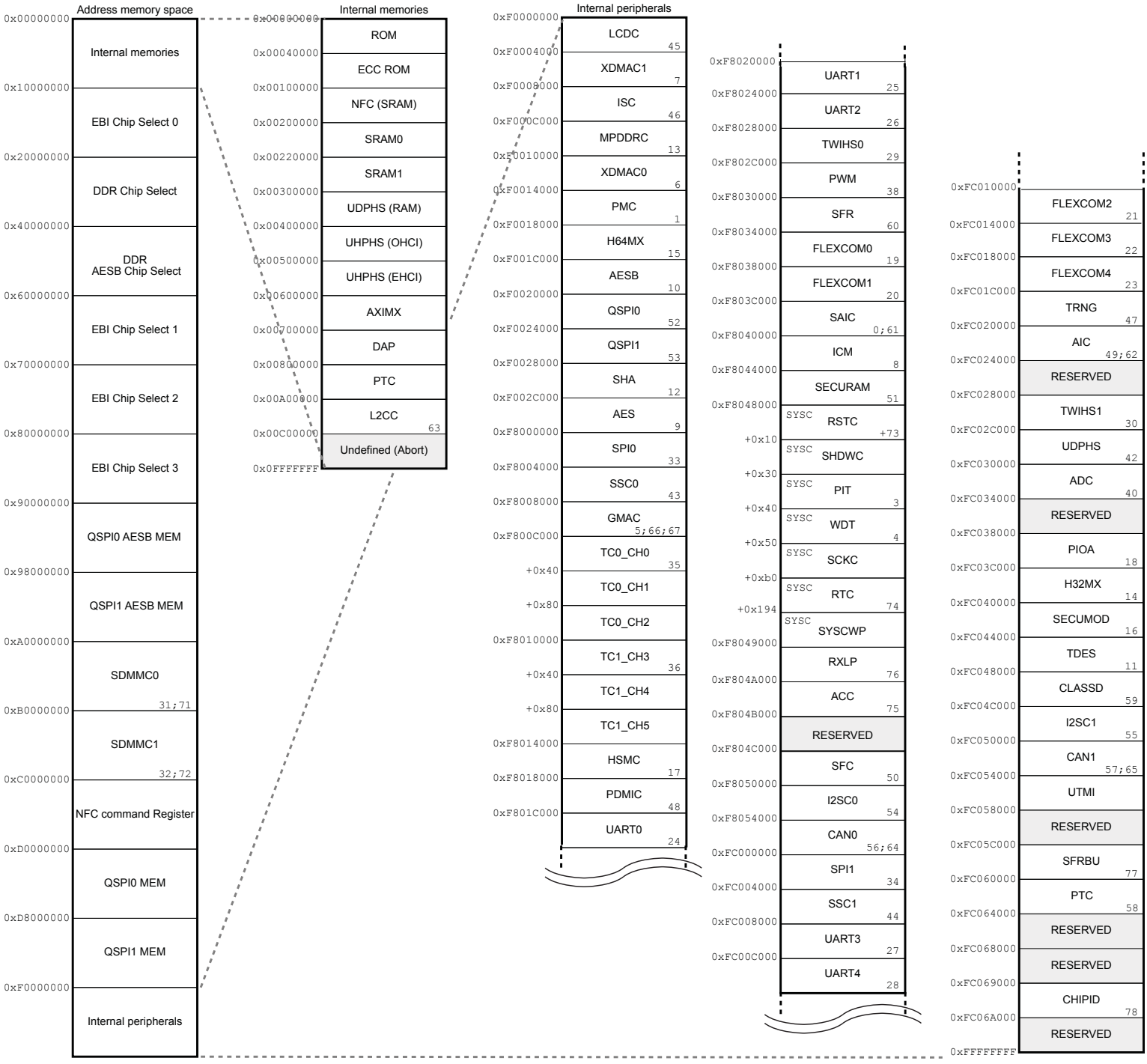
| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $t_1$ | Group 2 to Group 1 delay | Delay from the last Group 2 established[(1)] supply to the first Group1 supply turn-on | 1 | – | ms |
| $t_2$ | Group 1 delay[(2)] | Delay from the first group 1 established supply to the last Group 1 established supply | – | 1 | |
| $t_{RSTPU}$ | Reset delay at power-up | From the last established supply to NRST high. | 1 | – | |

**Note:**

1. An “established” supply refers to a power supply established at 90% of its final value.
2. Also applies to VDDIODDR when considered as part of Group 1.

# 8. Memories

### Figure 8-1. Memory Mapping



## 8.1 Embedded Memories

### 8.1.1 Internal SRAM

The SAMA5D2 embeds a total of 128 Kbytes of high-speed SRAM. After reset, and until the Remap command is performed, the SRAM is accessible at address 0x0020 0000. When the AXI Bus Matrix is remapped, the SRAM is also available at address 0x0.

The device features a second 128-Kbyte SRAM that can be allocated either to the L2 cache controller or used as an internal SRAM. After reset, this block is connected to the system SRAM, making the two 128-Kbyte RAMs contiguous. The SRAM_SEL bit, located in the SFR_L2CC_HRAMC register, is used to reassign this memory as a L2 cache memory.

### 8.1.2 Internal ROM

The product embeds one 160-Kbyte secured internal ROM mapped at address 0 after reset. The ROM contains a standard and secure bootloader as well as the BCH (Bose, Chaudhuri and Hocquenghem) code tables for NAND Flash ECC correction. The memory area containing the secure boot is automatically hidden after the execution of the secure boot while the one containing the code tables for ECC remains visible.

### 8.1.3 Boot Strategies

For standard boot strategies, refer to the section Standard Boot Strategies.

For secure boot strategies, refer to the document “SAMA5D2x Secure Boot Strategy”, document no. 44040 (Non-Disclosure Agreement required).

## 8.2 External Memory

The SAMA5D2 offers connections to a wide range of external memories or to parallel peripherals.

### 8.2.1 External Bus Interface

The External Bus Interface (EBI) is a 16-bit-wide interface working at MCK/2.

The EBI supports:

- Static memories
- 8-bit NAND Flash with 32-bit BCH ECC
- 16-bit NAND Flash

EBI I/Os accept three drive levels (Low, Medium, High) to avoid overshoots and provide the best performances according to the bus load and external memories voltage.

The drive levels are configured with the DRVSTR field in the PIO Configuration Register (PIO_CFGRx) if the corresponding line is nonsecure or the Secure PIO Configuration Register (S_PIO_CFGRx) if the I/O line is secure.

At reset, the selected drive is low. The user must make sure to program the correct drive according to the device load. The I/O embeds serial resistors for impedance matching.

### 8.2.2 Supported Memories on DDR Interface

- 16-bit or 32-bit external interface
- 512 Mbytes of address space on DDR CS and DDR/AES CS in 32-bit mode

- 256 Mbytes of address space on DDR CS and DDR/AES CS in 16-bit mode
- Supports 16-bit or 32-bit 8-bank DDR2, DDR3, LPDDR1, LPDDR2 and LPDDR3 memories
- Automatic drive level control
- Multiport
- Scramblable data path
- Port 0 of this interface has an embedded automatic AES encryption and decryption mechanism (refer to the section Advanced Encryption Standard Bridge (AESB)). Writing to or reading from the address 0x40000000 may trigger the encryption and decryption mechanism depending on the AESB on External Memories configuration.
- TrustZone: The multiport feature of this interface implies TrustZone configuration constraints. Refer to the section TrustZone Technology for more details.

### 8.2.3 Supported Memories on Static Memories and NAND Flash Interfaces

The Static Memory Controller is dedicated to interfacing external memory devices:

- Asynchronous SRAM-like memories and parallel peripherals
- NAND Flash (MLC and SLC) 8-bit datapath

The Static Memory Controller is able to drive up to four chip select. NCS3 is dedicated to the NAND Flash control.

The HSMC embeds a NAND Flash Controller (NFC). The NFC can handle automatic transfers, sending the commands and address cycles to the NAND Flash and transferring the contents of the page (for read and write) to the NFC SRAM. It minimizes the processor overhead.

In order to improve overall system performance, the DATA phase of the transfer can be DMA-assisted. The static memory embeds the NAND Flash Error Correcting Code controller with the following features:

- Algorithm based on BCH codes
- Supports also SLC 1-bit (BCH 2-bit), SLC 4-bit (BCH 4-bit)
- Programmable Error Correcting Capability
  - 2-bit, 4-bit, 8-bit and 16-bit errors for 512 bytes/sector (4-Kbyte page)
  - 24-bit error for 1024 bytes/sector (8-Kbyte page)
- Programmable sector size: 512 bytes or 1024 bytes
- Programmable number of sectors per page: 1, 2, 4 or 8 blocks of data per page
- Programmable spare area size
- Supports spare area ECC protection
- Supports 8-Kbyte page size using 1024 bytes/sector and 4-Kbyte page size using 512 bytes/sector
- Error detection is interrupt-driven
- Provides hardware acceleration for error location
- Finds roots of error-locator polynomial
- Programmable number of roots

### 8.2.4 DDR and SDMMC I/Os Calibration

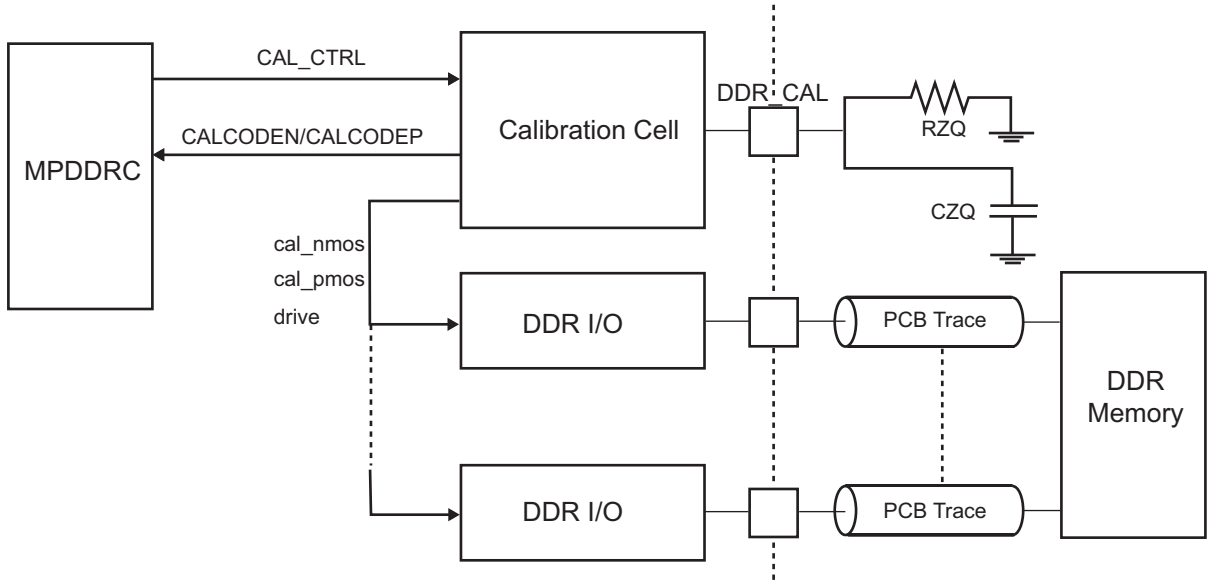
#### 8.2.4.1 DDR I/O Calibration

The DDR2/DDR3/LPDDR1/LPDDR2/LPDDR3/DDR3L I/Os embed an automatic impedance matching control to avoid overshoots and reach the best performance levels depending on the bus load and external memories. A serial termination connection scheme, where the driver has an output impedance matched to the characteristic impedance of the line, is used to improve signal quality and reduce EMI.

One specific analog input, DDR_CAL, is used to calibrate all DDR / IOs.

The MPDDRC supports the ZQ calibration procedure used to calibrate the SAMA5D2 DDR I/O drive strength and the commands to setup the external DDR device drive strength (refer to the section Multiport DDR-SDRAM Controller (MPDDRC)). The calibration cell supports all the memory types listed above.

**Figure 8-2. DDR Calibration Cell**




The calibration cell provides an input pin, DDR_CAL, loaded with one of the following resistor RZQ values:

- 24 KΩ for LPDDR2/LPDDR3
- 23 KΩ for DDR3L
- 22 KΩ for DDR3
- 21 KΩ for DDR2/LPDDR1

The typical value for CZQ is 22 pF.

#### 8.2.4.1.1 LPDDR2 Power Fail Management

The DDR controller (MPDDRC) is used to manage the LPDDR memory when an uncontrolled power off occurs.

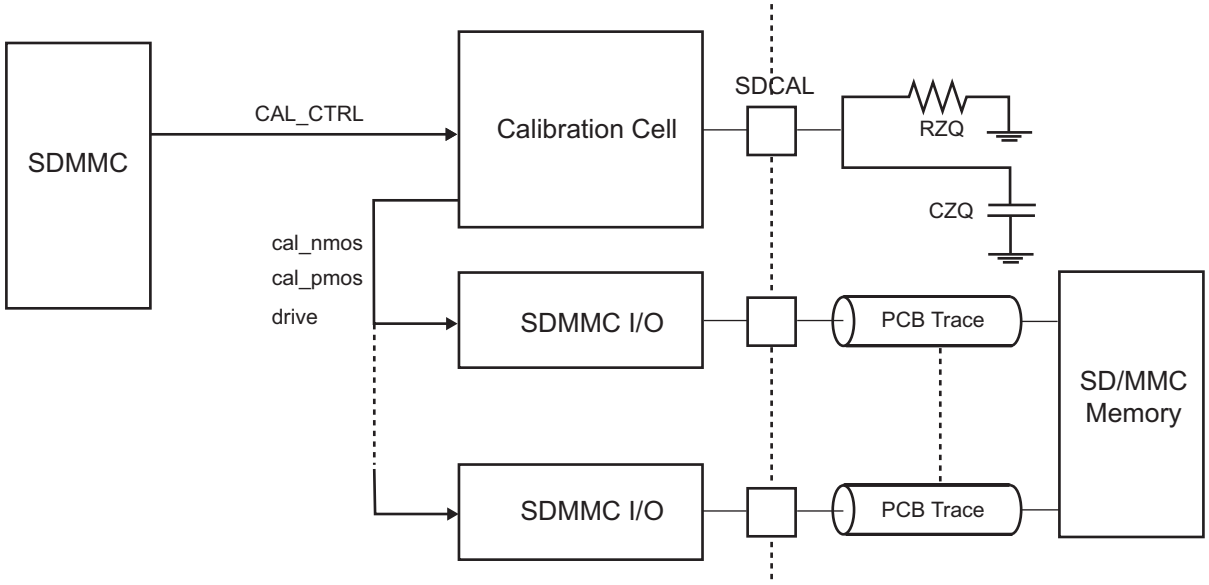
The DDR power rail must be monitored externally and generate an interrupt when a power fail condition is triggered. The interrupt handler must apply the sequence defined in the MPDDRC Low-power register (MPDDRC_LPR) by setting bit LPDDR2_PWOFF (LPDDR2 Power Off bit).

### 8.2.4.2 SDMMC I/O Calibration

The SAMA5D2 also embeds an SDMMC I/O calibration cell. The purpose of this block is to provide to e.MMC/SD I/Os an output impedance reference to limit the impact of process, voltage and temperature on the drivers output impedance. The impedance control is required at high frequency in order to improve signal quality.

The control and procedure to setup the SDMMC calibration cell is described in the section Secure Digital MultiMedia Card Controller (SDMMC).

**Figure 8-3. SDMMC I/O Calibration Cell**




The calibration cell provides an input pin SDCAL loaded with a 20 KΩ resistor for 1.8V memories and a 16.9 KΩ resistor for 3.3V memories.

According to the e.MMC specification, the output impedance calibration is mandatory for HS200 mode (1.8V) when it is not for other modes (3.3V).

In addition, according to the SD specification, the output impedance calibration is mandatory for 1.8V signaling when it is not for 3.3V signaling.

Thus, the calibration cell design is oriented to get the highest accuracy under 1.8V.

In case of interfacing which would need to operate under both 1.8V and 3.3V, external devices RZQ and CZQ must get values related to the 1.8V mode. The typical value for CZQ is 22 pF.

# 9. Event System

The events generated by peripherals are designed to be directly routed to peripherals managing/using these events without processor intervention. Peripherals receiving events contain logic by which to select the one required.

## 9.1 Real-time Event List

- Timers, PWM, and IO peripherals generate event triggers which are directly routed to event managers such as ADC, for example, to start measurement/conversion without processor intervention.
- ADC is connected to nine trigger inputs defined as two groups:
  - One group of eight elements for Timer Counter (TC0 to TC4), ADTRIG and PMW0 event0, PWM0 event1
  - One group of one element for low-rate trigger, RTC
- UART, USART, SPI, TWI, PWM, CLASSD, AES, SHA, ADC, PIO, TIMER (Capture mode) generate event triggers directly connected to DMA controllers (XDMAC) for data transfer without processor intervention.
- PWM safety events (faults) are in combinational form and directly routed from event generators (ADC, ACC, PMC, TIMER) to the PWM module.
- PWM receives external triggers to provide PFC, DC/DC functions.
- PWM output comparators generate events directly connected to TIMER.
- PMC safety event (clock failure detection) can be programmed to switch the MCK on a reliable main RC internal clock without processor intervention.

## 9.2 Real-time Event Mapping

**Table 9-1. Real-time Event Mapping List**

| Function | Application | Description | Event Source | Event Destination |
|---|---|---|---|---|
| Safety | General-purpose | Automatic switch to reliable main RC oscillator in case of main crystal clock failure(1) | Power Management Controller (PMC) | PMC |
| | General-purpose, motor control, power factor correction (PFC) | Puts the PWM outputs in Safe mode (main crystal clock failure detection)(1)(2) | | PWM |
| | Motor control, PFC | Puts the PWM outputs in Safe mode (overspeed, overcurrent detection, etc.)(2)(3) | ADC | |
| | Motor control | Puts the PWM outputs in Safe mode (overspeed detection through TIMER quadrature decoder)(2)(4) | Timer Counter Block (TC 0, 1, 2) | |
| | | | Timer Counter Block (TC 3, 4, 5) | |
| | General-purpose | Puts the PWM outputs in Safe mode (general-purpose fault inputs)(2) | 2 IOs (PWM_Flx) | |

.....continued

| Function | Application | Description | Event Source | Event Destination |
|---|---|---|---|---|
| Measurement trigger | General-purpose | Programmable delay in PWM(7) | PWM Event Line 0 | ADC |
| | | | PWM Event Line 1 | |
| | General-purpose | Trigger source selection in ADC(5) | IO (ADC_ADTRG) | |
| | | | TC Output 0 | |
| | | | TC Output 1 | |
| | | | TC Output 2 | |
| | | | TC Output 3 | |
| | | | TC Output 4 | RTCOUT0 |
| | General-purpose | Low-speed measurement(6) | RTC | RTCOUT1 |
| GTSUCOMP synchronous clock generation trigger | Audio | Trigger source selection in TC | GMAC GTSUCOMP Line | TC5 |
| Delay measurement | Motor control | Delay measurement between PWM outputs and TC inputs externally connected to power transistor bridge driver.(8)(9) | PWM Compare Line 0 | TC Input (A/B) 0 |
| | | | PWM Compare Line 1 | TC Input (A/B) 1 |
| | | | PWM Compare Line 2 | TC Input (A/B) 2 |

**Note:**

1. Refer to Main Crystal Oscillator Failure Detection in section Power Management Controller (PMC).
2. Refer to Fault Inputs and Fault Protection in section Pulse Width Modulation Controller (PWM).
3. Refer to Fault Outputin section Analog-to-Digital Converter (ADC).
4. Refer to 53.6.18 Fault Mode in section Timer Counter (TC).
5. Refer to ADC Trigger Register in section Analog-to-Digital Converter (ADC).
6. Refer to Waveform Generation in section Real-time Clock (RTC).
7. Refer to PWM Comparison Units and PWM Event Lines in section Pulse Width Modulation Controller (PWM).
8. Refer to 53.6.14 Synchronization with PWM in section Timer Counter (TC).
9. Refer to 55.6.2.2 Comparator in section Pulse Width Modulation Controller (PWM).

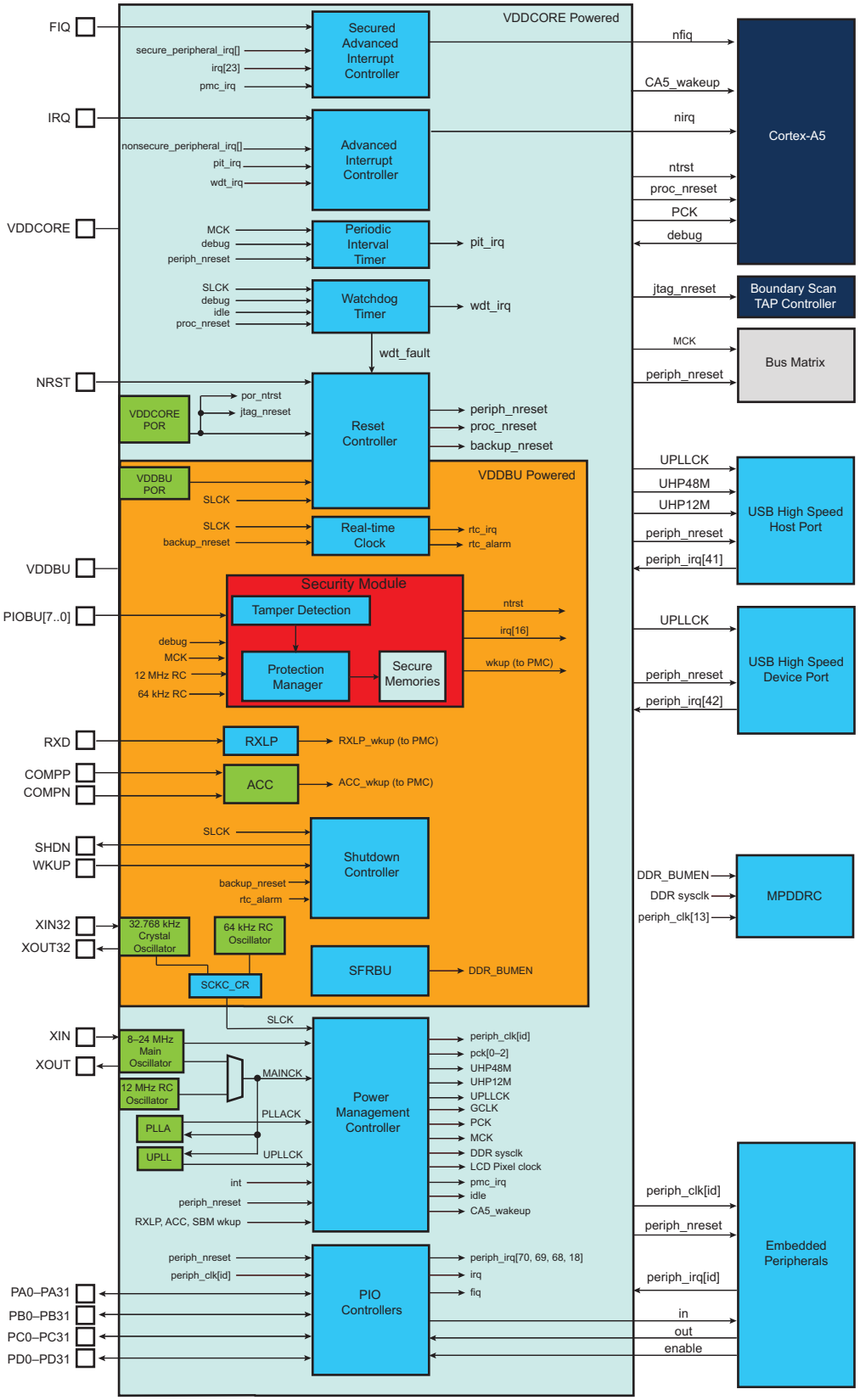
# 10. System Controller

The system controller is a set of peripherals handling key elements of the system, such as power, resets, clocks, time, interrupts, watchdog, etc.

The system controller's peripherals are all mapped between addresses 0xF8049000 and 0xF8048000.

The figure below shows the system controller block diagram.

**Figure 10-1. System Controller Block Diagram**



## 10.1 Power-On Reset

The SAMA5D2 embeds several Power-On Resets (PORs) to ensure the power supply is established when the reset is released. These PORs are dedicated to monitoring VDDBU, VDDIOP and VDDCORE respectively.

# 11. Peripherals

## 11.1 Peripheral Mapping

As shown in the figure Memory Mapping, the peripherals are mapped in the upper 256 Mbytes of the address space, between addresses 0xF000 0000 and 0xFFFC 0000.

## 11.2 Peripheral Identifiers

For details, refer to Table 18-9.

## 11.3 Peripheral Signal Multiplexing on I/O Lines

The SAMA5D2 features several PIO Controllers that multiplex the I/O lines of the peripheral set.

The table Pin Description (all packages) defines how the I/O lines are multiplexed on the different PIO Controllers. Several I/O sets are available for each peripheral. However, selecting I/Os from different I/O sets for one peripheral is prohibited.

The column “Reset State” shows whether the PIO line resets in I/O mode or in Peripheral mode. If I/O is shown, the PIO line resets in input with the pull-up enabled, so that the device is maintained in a static state as soon as the reset is released. As a result, the bit corresponding to the PIO line in register PIO_CFGR (PIO Configuration Register) resets low.

If a signal name is shown in the “Reset State” column, the PIO line is assigned to this function and the corresponding bit in PIO_CFGR resets high. That is the case for pins controlling memories, in particular address lines, which require the pin to be driven as soon as the reset is released.

The PIO state can be retained when the system enters in Backup mode.

## 11.4 Peripheral Clock Types

The table below lists the clock types available on embedded peripherals in SAMA5D2. Clock type suffixes HS and LS refer to Matrix (H64MX) and Matrix (H32MX), respectively.

For details on embedded peripherals, refer to the section Matrix (H64MX/H32MX).

**Table 11-1. Peripheral Clock Types**

| Clock Type | Description |
|---|---|
| HCLOCK_HS<br>HCLOCK_LS | AHB Clock.<br>Managed with the PMC_PCER, PMC_PCDR, PMC_PCSR and PMC_PCR registers of Peripheral Clock.(1) |
| PCLOCK_HS<br>PCLOCK_LS | APB Clock.<br>Managed with the PMC_PCER, PMC_PCDR, PMC_PCSR and PMC_PCR registers of Peripheral Clock. (1) |
| SYS_CLK_LS | This clock cannot be disabled. (2) |
| SYS_CLOCK | This clock cannot be disabled. (2) |

| .....continued | |
|---|---|
| **Clock Type** | **Description** |
| PROC_CLK | The clock related to Processor Clock (PCK) and managed with the PMC_SCDR and PMC_SCSR registers of PMC System Clock |
| SLOW_CLOCK | The clock related to the backup area and the RTC and managed with the SCKC_CR. This clock can be generated either by an external 32.768 kHz crystal oscillator or by the on-chip 64 kHz RC oscillator. |

**Note:**

1. Refer to the figure General Clock Block Diagram in the section Power Management Controller (PMC).
2. Refer to the MCK clock in the figure General Clock Block Diagram in the section Power Management Controller (PMC).

# 12. Chip Identifier (CHIPID)

## 12.1 Description

Chip Identifier (CHIPID) registers are used to recognize the device and its revision. These registers provide the sizes and types of the on-chip memories, as well as the set of embedded peripherals.

Two CHIPID registers are embedded: Chip ID Register (CHIPID_CIDR) and Chip ID Extension Register (CHIPID_EXID). Both registers contain a hard-wired value that is read-only.

The CHIPID_CIDR register contains the following fields:

- VERSION: Identifies the revision of the silicon
- EPROC: Indicates the embedded ARM processor
- NVPTYP and NVPSIZ: Identify the type of embedded non-volatile memory and the size
- SRAMSIZ: Indicates the size of the embedded SRAM
- ARCH: Identifies the set of embedded peripherals
- EXT: Shows the use of the extension identifier register

The CHIPID_EXID register is device-dependent and reads 0 if CHIPID_CIDR.EXT = 0.

## 12.2 Embedded Characteristics

- Chip ID Registers
  - Identification of the Device Revision, Sizes of the Embedded Memories, Set of Peripherals, Embedded Processor

**Table 12-1. SAMA5D2 Chip ID Registers**

| Chip Name | CHIPID_CIDR | CHIPID_EXID |
|---|---|---|
| ATSAMA5D22A-CU | 0x8A5C08C0 | 0x00000059 |
| ATSAMA5D24A-CU | 0x8A5C08C0 | 0x00000014 |
| ATSAMA5D27A-CU | 0x8A5C08C0 | 0x00000011 |
| ATSAMA5D28A-CU | 0x8A5C08C0 | 0x00000010 |
| ATSAMA5D21B-CU | 0x8A5C08C1 | 0x0000005A |
| ATSAMA5D22B-CN | 0x8A5C08C1 | 0x00000069 |
| ATSAMA5D22B-CU | 0x8A5C08C1 | 0x00000059 |
| ATSAMA5D23B-CN | 0x8A5C08C1 | 0x00000068 |
| ATSAMA5D23B-CU | 0x8A5C08C1 | 0x00000058 |
| ATSAMA5D24B-CU | 0x8A5C08C1 | 0x00000014 |
| ATSAMA5D26B-CN | 0x8A5C08C1 | 0x00000022 |
| ATSAMA5D26B-CU | 0x8A5C08C1 | 0x00000012 |
| ATSAMA5D27B-CN | 0x8A5C08C1 | 0x00000021 |

.....continued

| Chip Name | CHIPID_CIDR | CHIPID_EXID |
|---|---|---|
| ATSAMA5D27B-CU | 0x8A5C08C1 | 0x00000011 |
| ATSAMA5D28B-CN | 0x8A5C08C1 | 0x00000020 |
| ATSAMA5D28B-CU | 0x8A5C08C1 | 0x00000010 |
| ATSAMA5D21C-CU | 0x8A5C08C2 | 0x0000005A |
| ATSAMA5D22C-CN | 0x8A5C08C2 | 0x00000069 |
| ATSAMA5D22C-CU | 0x8A5C08C2 | 0x00000059 |
| ATSAMA5D23C-CN | 0x8A5C08C2 | 0x00000068 |
| ATSAMA5D23C-CU | 0x8A5C08C2 | 0x00000058 |
| ATSAMA5D24C-CU | 0x8A5C08C2 | 0x00000014 |
| ATSAMA5D26C-CN | 0x8A5C08C2 | 0x00000022 |
| ATSAMA5D26C-CU | 0x8A5C08C2 | 0x00000012 |
| ATSAMA5D27C-CN | 0x8A5C08C2 | 0x00000021 |
| ATSAMA5D27C-CU | 0x8A5C08C2 | 0x00000011 |
| ATSAMA5D28C-CN | 0x8A5C08C2 | 0x00000020 |
| ATSAMA5D28C-CU | 0x8A5C08C2 | 0x00000010 |

## 12.3 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CHIPID_CIDR | 7:0 | EPROC[2:0] | | | VERSION[4:0] | | | | |
| | | 15:8 | NVPSIZ2[3:0] | | | | NVPSIZ[3:0] | | | |
| | | 23:16 | ARCH[3:0] | | | | SRAMSIZ[3:0] | | | |
| | | 31:24 | EXT | NVPTYP[2:0] | | | ARCH[7:4] | | | |
| 0x04 | CHIPID_EXID | 7:0 | EXID[7:0] | | | | | | | |
| | | 15:8 | EXID[15:8] | | | | | | | |
| | | 23:16 | EXID[23:16] | | | | | | | |
| | | 31:24 | EXID[31:24] | | | | | | | |

### 12.3.1 Chip ID Register

**Name:** CHIPID_CIDR
**Offset:** 0x0
**Reset:** –
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | EXT | NVPTYP[2:0] | | | ARCH[7:4] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ARCH[3:0] | | | | SRAMSIZ[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NVPSIZ2[3:0] | | | | NVPSIZ[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EPROC[2:0] | | | VERSION[4:0] | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

**Bit 31 – EXT** Extension Flag

| Value | Description |
|---|---|
| 0 | Chip ID has a single register definition without extension. |
| 1 | An extended Chip ID exists. |

**Bits 30:28 – NVPTYP[2:0]** Nonvolatile Program Memory Type

| Value | Name | Description |
|---|---|---|
| 0 | ROM | ROM |
| 1 | ROMLESS | ROMless or on-chip Flash |
| 2 | FLASH | Embedded Flash Memory |
| 3 | ROM_FLASH | ROM and Embedded Flash Memory<br>– NVPSIZ is ROM size<br>– NVPSIZ2 is Flash size |
| 4 | SRAM | SRAM emulating ROM |

**Bits 27:20 – ARCH[7:0]** Architecture Identifier

| Value | Name | Description |
|---|---|---|
| 0xA5 | SAMA5 | SAMA5 |

**Bits 19:16 – SRAMSIZ[3:0]** Internal SRAM Size

| Value | Name | Description |
|---|---|---|
| 0 | 48K | 48 Kbytes |
| 1 | 192K | 192 Kbytes |
| 2 | 384K | 384 Kbytes |
| 3 | 6K | 6 Kbytes |
| 4 | 24K | 24 Kbytes |
| 5 | 4K | 4 Kbytes |
| 6 | 80K | 80 Kbytes |
| 7 | 160K | 160 Kbytes |
| 8 | 8K | 8 Kbytes |
| 9 | 16K | 16 Kbytes |
| 10 | 32K | 32 Kbytes |
| 11 | 64K | 64 Kbytes |
| 12 | 128K | 128 Kbytes |
| 13 | 256K | 256 Kbytes |
| 14 | 96K | 96 Kbytes |
| 15 | 512K | 512 Kbytes |

**Bits 15:12 – NVPSIZ2[3:0]** Second Nonvolatile Program Memory Size

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | 8K | 8 Kbytes |
| 2 | 16K | 16 Kbytes |
| 3 | 32K | 32 Kbytes |
| 4 | – | Reserved |
| 5 | 64K | 64 Kbytes |
| 6 | – | Reserved |
| 7 | 128K | 128 Kbytes |
| 8 | – | Reserved |
| 9 | 256K | 256 Kbytes |
| 10 | 512K | 512 Kbytes |
| 11 | – | Reserved |
| 12 | 1024K | 1024 Kbytes |
| 13 | – | Reserved |
| 14 | 2048K | 2048 Kbytes |
| 15 | – | Reserved |

**Bits 11:8 – NVPSIZ[3:0]** Nonvolatile Program Memory Size

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | 8K | 8 Kbytes |
| 2 | 16K | 16 Kbytes |
| 3 | 32K | 32 Kbytes |
| 4 | - | Reserved |
| 5 | 64K | 64 Kbytes |

| Value | Name | Description |
|---|---|---|
| 6 | - | Reserved |
| 7 | 128K | 128 Kbytes |
| 8 | 160K | 160 Kbytes |
| 9 | 256K | 256 Kbytes |
| 10 | 512K | 512 Kbytes |
| 11 | - | Reserved |
| 12 | 1024K | 1024 Kbytes |
| 13 | - | Reserved |
| 14 | 2048K | 2048 Kbytes |
| 15 | - | Reserved |

**Bits 7:5 – EPROC[2:0]** Embedded Processor

| Value | Name | Description |
|---|---|---|
| 0 | SAM x7 | Cortex-M7 |
| 1 | ARM946ES | ARM946ES |
| 2 | ARM7TDMI | ARM7TDMI |
| 3 | CM3 | Cortex-M3 |
| 4 | ARM920T | ARM920T |
| 5 | ARM926EJS | ARM926EJS |
| 6 | CA5 | Cortex-A5 |
| 7 | CM4 | Cortex-M4 |

**Bits 4:0 – VERSION[4:0]** Version of the Device
Current version of the device.

### 12.3.2 Chip ID Extension Register

**Name:** CHIPID_EXID
**Offset:** 0x4
**Reset:** –
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | EXID[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | EXID[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EXID[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EXID[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

**Bits 31:0 – EXID[31:0]** Chip ID Extension
This field is cleared if CHIPID_CIDR.EXT = 0.

# 13. Cortex-A5 Processor (ARM)

## 13.1 Reference Documents

The following table gives the references of the documents and their denominations in this document.

**Table 13-1. Reference Documents**

| Document Title | Document No. |
|---|---|
| Cortex-A5 Technical Reference Manual | ARM DDI 0433 |
| Cortex-A5 Floating-Point Unit Technical Reference Manual | ARM DDI 0449 |
| Cortex-A5 NEON Media Processing Engine Technical Reference Manual | ARM DDI 0450 |
| ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition | ARM DDI 0406 |

## 13.2 Description

The ARM Cortex-A5 processor is a high-performance, low-power, ARM macrocell with an L1 cache subsystem that provides full virtual memory capabilities. The Cortex-A5 processor implements the ARMv7 architecture and runs 32-bit ARM instructions, 16-bit and 32-bit Thumb-2 instructions, and 8-bit Java™ byte codes in Jazelle® state.

The Cortex-A5 NEON Media Processing Engine (MPE) extends the Cortex-A5 functionality to provide support for the ARM v7 Advanced SIMD v2 and Vector Floating-Point v4 (VFPv4) instruction sets. The Cortex-A5 NEON MPE provides flexible and powerful acceleration for signal processing algorithms including multimedia such as image processing, video decode/encode, 2D/3D graphics, and audio. See the Cortex-A5 NEON Media Processing Engine Technical Reference Manual.

The Cortex-A5 processor includes TrustZone® technology to enhance security by partitioning the SoC's hardware and software resources in a Secure world for the security subsystem and a Normal world for the rest, enabling a strong security perimeter to be built between the two. See Security Extensions overview in the Cortex-A5 Technical Reference Manual. See the ARM Architecture Reference Manual for details on how TrustZone works in the architecture.

**Note:** All ARM publications referenced in this datasheet can be found at www.arm.com.

### 13.2.1 Power Management

The Cortex-A5 design supports the following main levels of power management:

- Run Mode
- Standby Mode

#### 13.2.1.1 Run Mode

Run mode is the normal mode of operation where all of the processor functionality is available. Everything, including core logic and embedded RAM arrays, is clocked and powered up.

#### 13.2.1.2 Standby Mode

Standby mode disables most of the clocks of the processor, while keeping it powered up. This reduces the power drawn to the static leakage current, plus a small clock power overhead required to enable the processor to wake up from Standby mode. The transition from Standby mode to Run mode is caused by one of the following:

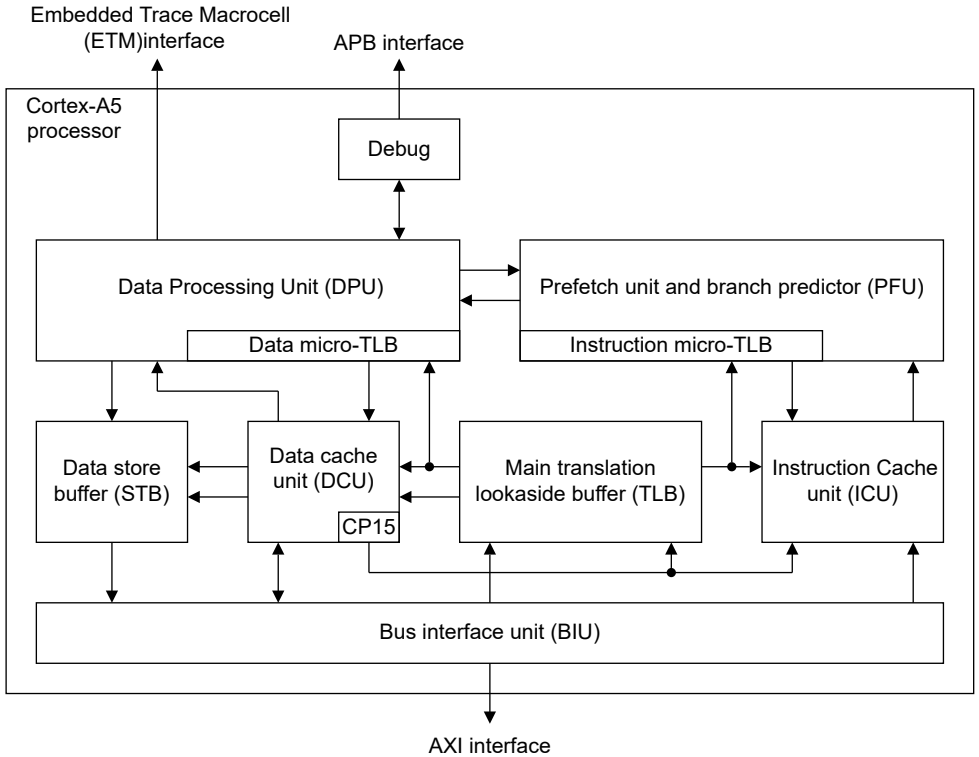
- the arrival of an interrupt, either masked or unmasked
- the arrival of an event, if standby mode was initiated by a Wait for Event (WFE) instruction
- a debug request, when either debug is enabled or disabled
- a reset.

## 13.3 Embedded Characteristics

- In-order Pipeline with Dynamic Branch Prediction
- ARM, Thumb-2, and Thumb-2EE Instruction Set Support
- TrustZone Security Extensions
- Harvard Level 1 Memory System with a Memory Management Unit (MMU)
- 32 Kbytes Data Cache
- 32 Kbytes Instruction Cache
- 64-bit AXI Master Interface
- ARM v7 Debug Architecture
- Trace Support through an Embedded Trace Macrocell (ETM) Interface
- Media Processing Engine (MPE) with NEON Technology
- Jazelle Hardware Acceleration

## 13.4 Block Diagram

**Figure 13-1. Cortex-A5 Processor Top-level Diagram**



## 13.5 Programmer Model

### 13.5.1 Processor Operating Modes

The following operation modes are present in all states:

- User mode (USR) is the usual ARM program execution state. It is used for executing most application programs.
- Fast Interrupt (FIQ) mode is used for handling fast interrupts. It is suitable for high-speed data transfer or channel process.
- Interrupt (IRQ) mode is used for general-purpose interrupt handling.
- Supervisor mode (SVC) is a protected mode for the operating system.
- Abort mode (ABT) is entered after a data or instruction prefetch abort.
- System mode (SYS) is a privileged user mode for the operating system.
- Undefined mode (UND) is entered when an undefined instruction exception occurs.
- Monitor mode (MON) is secure mode that enables change between Secure and Non-secure states, and can also be used to handle any of FIQs, IRQs and external aborts. Entered on execution of a Secure Monitor Call (SMC) instruction.

Mode changes may be made under software control, or may be brought about by external interrupts or exception processing. Most application programs execute in User Mode. The non-user modes, known as privileged modes, are entered in order to service interrupts or exceptions or to access protected resources.

### 13.5.2 Processor Operating States

The processor has the following instruction set states controlled by the T bit and J bit in the CPSR.

- ARM state:
  The processor executes 32-bit, word-aligned ARM instructions.
- Thumb-2 state:
  The processor executes 16-bit and 32-bit, halfword-aligned Thumb-2 instructions.
- Thumb-2EE state:
  The processor executes a variant of the Thumb-2 instruction set designed as a target for dynamically generated code. This is code compiled on the device either shortly before or during execution from a portable bytecode or other intermediate or native representation.
- Jazelle state:
  The processor executes variable length, byte-aligned Java bytecodes.

The J bit and the T bit determine the instruction set used by the processor. The table below shows the encoding of these bits.

**Table 13-2. CPSR J and T Bit Encoding**

| J | T | Instruction Set State |
|---|---|---|
| 0 | 0 | ARM |
| 0 | 1 | Thumb-2 |
| 1 | 0 | Jazelle |
| 1 | 1 | Thumb-2EE |

Alternating between ARM and Thumb-2 states does not affect the processor mode or the register contents. See the ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition for information on entering and exiting Thumb-2EE state.

#### 13.5.2.1 Switching State

It is possible to change the instruction set state of the processor between:

- ARM state and Thumb-2 state using the BX and BLX instructions.
- Thumb-2 state and Thumb-2EE state using the ENTERX and LEAVEX instructions.
- ARM and Jazelle state using the BXJ instruction.
- Thumb-2 and Jazelle state using the BXJ instruction.

See the ARM Architecture Reference Manual for more information about changing instruction set state.

### 13.5.3 Cortex-A5 Registers

This view provides 16 ARM core registers, R0 to R15, that include the Stack Pointer (SP), Link Register (LR), and Program Counter (PC). The current execution mode determines the selected set of registers, as shown in the table below. This shows that the arrangement of the registers provides duplicate copies of some registers, with the current register selected by the execution mode. This arrangement is described as banking of the registers, and the duplicated copies of registers are referred to as banked registers.

**Table 13-3. Cortex-A5 Modes and Registers Layout**

| User and System | Monitor | Supervisor | Abort | Undefined | Interrupt | Fast Interrupt |
|---|---|---|---|---|---|---|
| R0 | R0 | R0 | R0 | R0 | R0 | R0 |
| R1 | R1 | R1 | R1 | R1 | R1 | R1 |
| R2 | R2 | R2 | R2 | R2 | R2 | R2 |
| R3 | R3 | R3 | R3 | R3 | R3 | R3 |
| R4 | R4 | R4 | R4 | R4 | R4 | R4 |
| R5 | R5 | R5 | R5 | R5 | R5 | R5 |
| R6 | R6 | R6 | R6 | R6 | R6 | R6 |
| R7 | R7 | R7 | R7 | R7 | R7 | R7 |
| R8 | R8 | R8 | R8 | R8 | R8 | R8_FIQ(1) |
| R9 | R9 | R9 | R9 | R9 | R9 | R9_FIQ(1) |
| R10 | R10 | R10 | R10 | R10 | R10 | R10_FIQ(1) |
| R11 | R11 | R11 | R11 | R11 | R11 | R11_FIQ(1) |
| R12 | R12 | R12 | R12 | R12 | R12 | R12_FIQ(1) |
| R13(1) | R13_MON(1) | R13_SVC(1) | R13_ABT(1) | R13_UND(1) | R13_IRQ(1) | R13_FIQ(1) |
| R14(1) | R14_MON(1) | R14_SVC(1) | R14_ABT(1) | R14_UND(1) | R14_IRQ(1) | R14_FIQ(1) |
| PC | PC | PC | PC | PC | PC | PC |

.....continued

| User and System | Monitor | Supervisor | Abort | Undefined | Interrupt | Fast Interrupt |
|---|---|---|---|---|---|---|
| CPSR | CPSR | CPSR | CPSR | CPSR | CPSR | CPSR |
| | SPSR_MON(1) | SPSR_SVC(1) | SPSR_ABT(1) | SPSR_UND(1) | SPSR_IRQ(1) | SPSR_FIQ(1) |

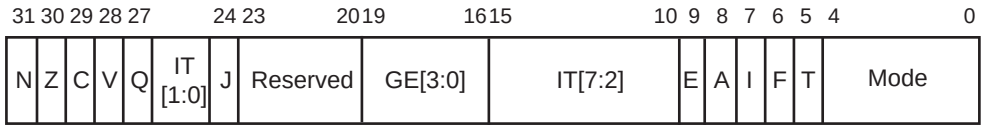
**Note:**

1. Mode-specific banked registers.

The core contains one CPSR, and six SPSRs for exception handlers to use. The program status registers:

- hold information about the most recently performed ALU operation
- control the enabling and disabling of interrupts
- set the processor operating mode

**Figure 13-2. Status Register Format**




- N: Negative, Z: Zero, C: Carry, and V: Overflow are the four ALU flags
- Q: cumulative saturation flag
- IT: If-Then execution state bits for the Thumb-2 IT (If-Then) instruction
- J: Jazelle bit, see the description of the T bit
- GE: Greater than or Equal flags, for SIMD instructions
- E: Endianness execution state bit. Controls the load and store endianness for data accesses. This bit is ignored by instruction fetches.
  - E = 0: Little endian operation
  - E = 1: Big endian operation
- A: Asynchronous abort disable bit. Used to mask asynchronous aborts.
- I: Interrupt disable bit. Used to mask IRQ interrupts.
- F: Fast interrupt disable bit. Used to mask FIQ interrupts.
- T: Thumb-2 execution state bit. This bit and the J execution state bit, bit [24], determine the instruction set state of the processor, ARM, Thumb-2, Jazelle, or Thumb-2EE.
- Mode: five bits to encode the current processor mode. The effect of setting M[4:0] to a reserved value is UNPREDICTABLE.

**Table 13-4. Processor Mode vs. Mode Field**

| Mode | M[4:0] |
|---|---|
| USR | 10000 |
| FIQ | 10001 |
| IRQ | 10010 |
| SVC | 10011 |
| MON | 10110 |

| .....continued | |
|---|---|
| **Mode** | **M[4:0]** |
| ABT | 10111 |
| UND | 11011 |
| SYS | 11111 |
| Reserved | Other |

#### 13.5.3.1 CP15 Coprocessor

Coprocessor 15, or System Control Coprocessor CP15, is used to configure and control all the items in the list below:

- Cortex-A5
- Caches (ICache, DCache and write buffer)
- MMU
- Security
- Other system options

To control these features, CP15 provides 16 additional registers. See the table below.

**Table 13-5. CP15 Registers**

| Register | Name | Read/Write |
|---|---|---|
| 0 | ID Code(1) | Read/Unpredictable |
| 0 | Cache type(1) | Read/Unpredictable |
| 1 | Control(1) | Read/Write |
| 1 | Security(1) | Read/Write |
| 2 | Translation Table Base | Read/Write |
| 3 | Domain Access Control | Read/Write |
| 4 | Reserved | None |
| 5 | Data fault Status(1) | Read/Write |
| 5 | Instruction fault status | Read/Write |
| 6 | Fault Address | Read/Write |
| 7 | Cache and MMU Operations(1) | Read/Write |
| 8 | TLB operations | Unpredictable/Write |
| 9 | Cache lockdown(1) | Read/Write |
| 10 | TLB lockdown | Read/Write |
| 11 | Reserved | None |
| 12 | Interrupts management | Read/Write |
| 12 | Monitor vectors | Read-only |

| .....continued | | |
|---|---|---|
| **Register** | **Name** | **Read/Write** |
| 13 | FCSE PID[1] | Read/Write |
| 13 | Context ID[1] | Read/Write |
| 14 | Reserved | None |
| 15 | Test configuration | Read/Write |

**Note:**

1. This register provides access to more than one register. The register accessed depends on the value of the CRm field or opcode_2 field.

### 13.5.4 CP15 Register Access

CP15 registers can only be accessed in privileged mode by:

- MCR (Move to Coprocessor from ARM Register) instruction is used to write an ARM register to CP15.
- MRC (Move to ARM Register from Coprocessor) instruction is used to read the value of CP15 to an ARM register.

Other instructions such as CDP, LDC, STC can cause an undefined instruction exception.

The assembler code for these instructions is:

```
MCR/MRC{cond} p15, opcode_1, Rd, CRn, CRm, opcode_2
```

The MCR/MRC instructions bit pattern is shown below:

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | cond | | | | 1 | 1 | 1 | 0 |
| **Bit** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| | opcode_1 | | | L | CRn | | | |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| | Rd | | | | 1 | 1 | 1 | 1 |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| | opcode_2 | | | 1 | CRm | | | |

**• CRm[3:0]: Specified Coprocessor Action**

Determines specific coprocessor action. Its value is dependent on the CP15 register used. For details, see CP15 specific register behavior.

**• opcode_2[7:5]**

Determines specific coprocessor operation code. By default, set to 0.

**• Rd[15:12]: ARM Register**

Defines the ARM register whose value is transferred to the coprocessor. If R15 is chosen, the result is unpredictable.

**• CRn[19:16]: Coprocessor Register**

Determines the destination coprocessor register.

**• L: Instruction Bit**

0: MCR instruction

1: MRC instruction

**• opcode_1[23:20]: Coprocessor Code**

Defines the coprocessor specific code. Value is c15 for CP15.

**• cond [31:28]: Condition**

### 13.5.5 Addresses in the Cortex-A5 Processor

The Cortex-A5 processor operates using virtual addresses (VAs). The Memory Management Unit (MMU) translates these VAs into the physical addresses (PAs) used to access the memory system. Translation tables hold the mappings between VAs and PAs.

See the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R* edition for more information.

When the Cortex-A5 processor is executing in Non-secure state, the processor performs translation table look-ups using the Non-secure versions of the Translation Table Base Registers. In this situation, any VA can only translate into a Non-secure PA. When it is in Secure state, the Cortex-A5 processor performs translation table look-ups using the Secure versions of the Translation Table Base Registers. In this situation, the security state of any VA is determined by the NS bit of the translation table descriptors for that address.

Following is an example of the address manipulation that occurs when the Cortex-A5 processor requests an instruction:

1. The Cortex-A5 processor issues the VA of the instruction as Secure or Non-secure VA accesses according to the state the processor is in.
2. The instruction cache is indexed by the bits of the VA. The MMU performs the translation table look-up in parallel with the cache access. If the processor is in the Secure state it uses the Secure translation tables, otherwise it uses the Non-secure translation tables.
3. If the protection check carried out by the MMU on the VA does not abort and the PA tag is in the instruction cache, the instruction data is returned to the processor.
4. If there is a cache miss, the MMU passes the PA to the AXI bus interface to perform an external access. The external access is always Non-secure when the core is in the Non-secure state. In the Secure state, the external access is Secure or Non-secure according to the NS attribute value in the selected translation table entry. In Secure state, both L1 and L2 translation table walk accesses are marked as Secure, even if the first level descriptor is marked as NS.

### 13.5.6 Security Extensions Overview

The purpose of the Security Extensions is to enable the construction of a secure software environment. See the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R* edition for details of the Security Extensions.

#### 13.5.6.1 System Boot Sequence

> ⚠ **CAUTION** The Security Extensions enable the construction of an isolated software environment for more secure execution, depending on a suitable system design around the processor. The technology does not protect the processor from hardware attacks, and care must be taken to be sure that the hardware containing the reset handling code is appropriately secure.

The processor always boots in the privileged Supervisor mode in the Secure state, with the NS bit set to 0. This means that code that does not attempt to use the Security Extensions always runs in the Secure state. If the software uses both Secure and Non-secure states, the less trusted software, such as a complex operating system and application code running under that operating system, executes in Non-secure state, and the most trusted software executes in the Secure state.

The following sequence is expected to be typical use of the Security Extensions:

1. Exit from reset in Secure state.
2. Configure the security state of memory and peripherals. Some memory and peripherals are accessible only to the software running in Secure state.
3. Initialize the secure operating system. The required operations depend on the operating system, and include initialization of caches, MMU, exception vectors, and stacks.
4. Initialize Secure Monitor software to handle exceptions that switch execution between the Secure and Non-secure operating systems.
5. Optionally lock aspects of the secure state environment against further configuration.
6. Pass control through the Secure Monitor software to the non-secure OS with an SMC instruction.
7. Enable the Non-secure operating system to initialize. The required operations depend on the operating system, and typically include initialization of caches, MMU, exception vectors, and stacks.

The overall security of the secure software depends on the system design, and on the secure software itself.

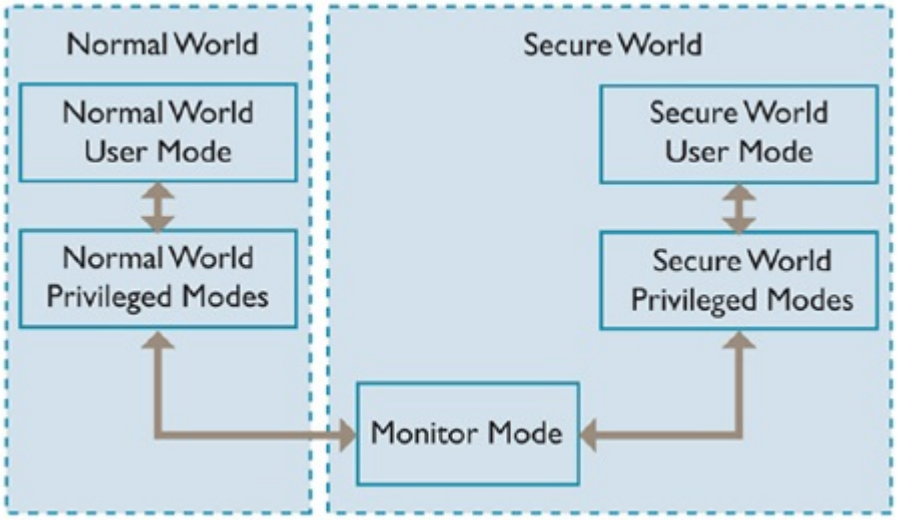
### 13.5.7 TrustZone

#### 13.5.7.1 Hardware

TrustZone enables a single physical processor core to execute code safely and efficiently from both the Normal world and the Secure world. This removes the need for a dedicated security processor core, saving silicon area and power, and allowing high performance security software to run alongside the Normal world operating environment.

The two virtual processors context switch via a new processor mode called monitor mode when changing the currently running virtual processor.

**Figure 13-3. TrustZone Hardware Implementation**

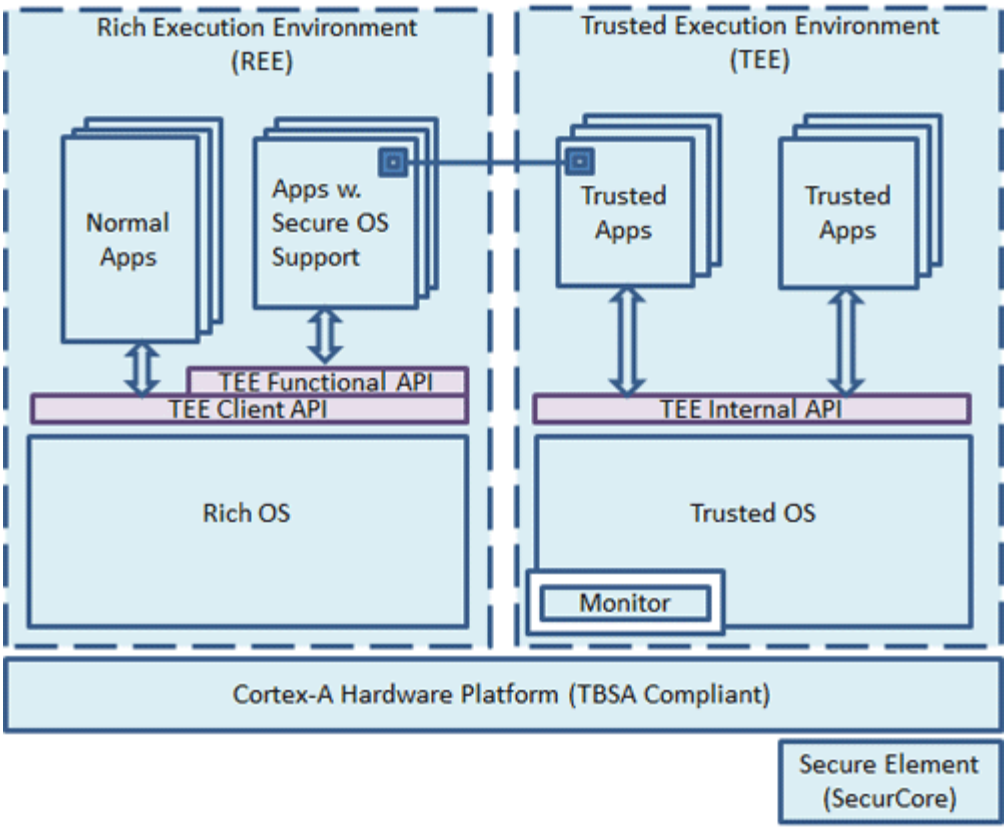



#### 13.5.7.2 Software

The mechanisms by which the physical processor can enter monitor mode from the Normal world are tightly controlled, and are all viewed as exceptions to the monitor mode software. Software executing a dedicated instruction can trigger entry to monitor, the Secure Monitor Call (SMC) instruction, or by a subset of the hardware exception mechanisms. Configuration of the IRQ, FIQ, external Data Abort, and external Prefetch Abort exceptions can cause the processor to switch into monitor mode.

The software that executes within monitor mode is implementation defined, but it generally saves the state of the current world and restores the state of the world at the location to which it switches. It then performs a return-from-exception to restart processing in the restored world.

**Figure 13-4. TrustZone Software Implementation in a Trusted Execution Environment (TEE)**




#### 13.5.7.3 Debug

TrustZone hardware architecture is a security-aware debug infrastructure that can enable control over access to secure world debug, without impairing debug visibility of the Normal world. This is controlled with bits in the Secure Fuse Controller.

**Note:** Secure debug modes are described in the document SAMA5D2 External Tamper Protections, ref. 44095. Contact a Microchip sales representative for further details.

## 13.6 Memory Management Unit (MMU)

### 13.6.1 About the MMU

The MMU works with the L1 and L2 memory system to translate virtual addresses to physical addresses. It also controls accesses to and from external memory.

The ARM v7 Virtual Memory System Architecture (VMSA) features include the following:

- Page table entries that support:
  - 16 Mbyte supersections. The processor supports supersections that consist of 16 Mbyte blocks of memory.
  - 1 Mbyte sections
  - 64 Kbyte large pages
  - 4 Kbyte small pages
- 16 access domains

- Global and application-specific identifiers to remove the requirement for context switch TLB flushes.
- Extended permissions checking capability.

TLB maintenance and configuration operations are controlled through a dedicated coprocessor, CP15, integrated with the core. This coprocessor provides a standard mechanism for configuring the L1 memory system.

See the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R* edition for a full architectural description of the ARMv7 VMSA.

### 13.6.2 Memory Management System

The Cortex-A5 processor supports the ARM v7 VMSA including the TrustZone security extension. The translation of a Virtual Address (VA) used by the instruction set architecture to a Physical Address (PA) used in the memory system and the management of the associated attributes and permissions is carried out using a two-level MMU.

The first level MMU uses a Harvard design with separate micro TLB structures in the PFU for instruction fetches (IuTLB) and in the DPU for data read and write requests (DuTLB).

A miss in the micro TLB results in a request to the main unified TLB shared between the data and instruction sides of the memory system. The TLB consists of a 128-entry two-way set-associative RAM based structure. The TLB page-walk mechanism supports page descriptors held in the L1 data cache. The caching of page descriptors is configured globally for each translation table base register, TTBRx, in the system coprocessor, CP15.

The TLB contains a hitmap cache of the page types which have already been stored in the TLB.

#### 13.6.2.1 Memory Types

Although various different memory types can be specified in the page tables, the Cortex-A5 processor does not implement all possible combinations:

- Write-through caches are not supported. Any memory marked as write-through is treated as Non-cacheable.
- The outer shareable attribute is not supported. Anything marked as outer shareable is treated in the same way as inner shareable.
- Write-back no write-allocate is not supported. It is treated as write-back write-allocate.

The table below shows the treatment of each different memory type in the Cortex-A5 processor in addition to the architectural requirements.

**Table 13-6. Treatment of Memory Attributes**

| Memory Type Attribute | Shareability | Other Attributes | Notes |
|---|---|---|---|
| Strongly Ordered | – | – | – |
| Device | Non-shareable | – | – |
| | Shareable | – | – |

.....continued

| Memory Type Attribute | Shareability | Other Attributes | Notes |
|---|---|---|---|
| Normal | Non-shareable | Non-cacheable | Does not access L1 caches |
| | | Write-through cacheable | Treated as non-cacheable |
| | | Write-back cacheable, write allocate | Can dynamically switch to no write allocate, if more than three full cache lines are written in succession |
| | | Write-back cacheable, no write allocate | Treated as non-shareable write-back cacheable, write allocate |
| | Inner shareable | Non-cacheable | – |
| | | Write-through cacheable | Treated as inner shareable non-cacheable |
| | | Write-back cacheable, write allocate | Treated as inner shareable non-cacheable unless the SMP bit in the Auxiliary Control Register is set (ACTLR[6] = b1). If this bit is set the area is treated as write-back cacheable write allocate. |
| | | Write-back cacheable, no write allocate | |
| | Outer shareable | Non-cacheable | Treated as inner shareable non-cacheable |
| | | Write-through cacheable | |
| | | Write-back cacheable, write allocate | Treated as inner shareable non-cacheable unless the SMP bit in the Auxiliary Control Register is set (ACTLR[6] = b1). If this bit is set the area is treated as write-back cacheable write allocate. |
| | | Write-back cacheable, no write allocate | |

### 13.6.3 Translation Lookaside Buffer (TLB) Organization

The Translation Lookaside Buffer (TLB) has two parts:

- Micro TLB
- Main TLB

#### 13.6.3.1 Micro TLB

The first level of caching for the page table information is a micro TLB of 10 entries that is implemented on each of the instruction and data sides. These blocks provide a lookup of the virtual addresses in a single cycle.

The micro TLB returns the physical address to the cache for the address comparison, and also checks the access permissions to signal either a Prefetch Abort or a Data Abort.

All main TLB related maintenance operations affect both the instruction and data micro TLBs, causing them to be flushed. In the same way, any change of the following registers causes the micro TLBs to be flushed:

- Context ID Register (CONTEXTIDR)
- Domain Access Control Register (DACR)
- Primary Region Remap Register (PRRR)
- Normal Memory Remap Register (NMRR)
- Translation Table Base Registers (TTBR0 and TTBR1)

#### 13.6.3.2 Main TLB

Misses from the instruction and data micro TLBs are handled by a unified main TLB. Accesses to the main TLB take a variable number of cycles, according to competing requests from each of the micro TLBs and other implementation-dependent factors.

The main TLB is 128-entry two-way set-associative.

**TLB match process**

Each TLB entry contains a virtual address, a page size, a physical address, and a set of memory properties. Each is marked as being associated with a particular application space (ASID), or as global for all application spaces. The CONTEXTIDR determines the currently selected application space.

A TLB entry matches when these conditions are true:

- Its virtual address matches that of the requested address.
- Its Non-secure TLB ID (NSTID) matches the Secure or Non-secure state of the MMU request.
- Its ASID matches the current ASID in the CONTEXTIDR or is global.

The operating system must ensure that, at most, one TLB entry matches at any time. The TLB can store entries based on the following block sizes:

| | |
|---|---|
| **Supersections** | Describe 16 Mbyte blocks of memory |
| **Sections** | Describe 1 Mbyte blocks of memory |
| **Large pages** | Describe 64 Kbyte blocks of memory |
| **Small pages** | Describe 4 Kbyte blocks of memory |

Supersections, sections and large pages are supported to permit mapping of a large region of memory while using only a single entry in the TLB. If no mapping for an address is found within the TLB, then the translation table is automatically read by hardware and a mapping is placed in the TLB.

### 13.6.4 Memory Access Sequence

When the processor generates a memory access, the MMU:

1. Performs a lookup for the requested virtual address and current ASID and security state in the relevant instruction or data micro TLB.
2. If there is a miss in the micro TLB, performs a lookup for the requested virtual address and current ASID and security state in the main TLB.
3. If there is a miss in main TLB, performs a hardware translation table walk.

The MMU can be configured to perform hardware translation table walks in cacheable regions by setting the IRGN bits in Translation Table Base Register 0 and Translation Table Base Register 1. If the encoding of the IRGN bits is write-back, an L1 data cache lookup is performed and data is read from the data

cache. If the encoding of the IRGN bits is write-through or non-cacheable, an access to external memory is performed. For more information, see *Cortex-A5 Technical Reference Manual*.

The MMU might not find a global mapping, or a mapping for the currently selected ASID, with a matching Non-secure TLB ID (NSTID) for the virtual address in the TLB. In this case, the hardware does a translation table walk if the translation table walk is enabled by the PD0 or PD1 bit in the Translation Table Base Control Register. If translation table walks are disabled, the processor returns a Section Translation fault. For more information, see *Cortex-A5 Technical Reference Manual*.

If the TLB finds a matching entry, it uses the information in the entry as follows:

1. The access permission bits and the domain determine if the access is enabled. If the matching entry does not pass the permission checks, the MMU signals a memory abort. See the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R* edition for a description of access permission bits, abort types and priorities, and for a description of the Instruction Fault Status Register (IFSR) and Data Fault Status Register (DFSR).
2. The memory region attributes specified in both the TLB entry and the CP15 c10 remap registers determine if the access is
   – Secure or Non-secure
   – Shared or not
   – Normal memory, Device, or Strongly-ordered

   For more information, see *Cortex-A5 Technical Reference Manual*, Memory region remap.
3. The TLB translates the virtual address to a physical address for the memory access.

### 13.6.5 Interaction with Memory System

The MMU can be enabled or disabled as described in the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition*.

### 13.6.6 External Aborts

External memory errors are defined as those that occur in the memory system rather than those that are detected by the MMU. External memory errors are expected to be extremely rare. External aborts are caused by errors flagged by the AXI interfaces when the request goes external to the Cortex-A5 processor. External aborts can be configured to trap to Monitor mode by setting the EA bit in the Secure Configuration Register. For more information, see *Cortex-A5 Technical Reference Manual*.

#### 13.6.6.1 External Aborts on Data Write

Externally generated errors during a data write can be asynchronous. This means that the r14_abt on entry into the abort handler on such an abort might not hold the address of the instruction that caused the exception.

The DFAR is Unpredictable when an asynchronous abort occurs.

Externally generated errors during data read are always synchronous. The address captured in the DFAR matches the address which generated the external abort.

#### 13.6.6.2 Synchronous and Asynchronous Aborts

The section System Control in the *Cortex-A5 Technical Reference Manual* describes synchronous and asynchronous aborts, their priorities, and the IFSR and DFSR. To determine a fault type, read the DFSR for a data abort or the IFSR for an instruction abort.

The processor supports an Auxiliary Fault Status Register for software compatibility reasons only. The processor does not modify this register because of any generated abort.

### 13.6.7 MMU Software Accessible Registers

The system control coprocessor registers, CP15, in conjunction with page table descriptors stored in memory, control the MMU.

Access all the registers with instructions of the form:

```
MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>
```

```
MCR p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>
```

CRn is the system control coprocessor register. Unless specified otherwise, CRm and opcode_2 should be zero.

# 14. L2 Cache Controller (L2CC)

## 14.1 Description

The L2 Cache Controller (L2CC) is based on the L2CC-PL310 ARM multi-way cache macrocell, version r3p2. The addition of an on-chip secondary cache, also referred to as a Level 2 or L2 cache, is a method of improving the system performance when significant memory traffic is generated by the processor.

## 14.2 Embedded Characteristics

- 8-Way Set Associative Cache Architecture
- Data Banking Not Supported
- No Parity Bit Embedded
- Lockdown by Master Not Supported
- Lockdown by Line Not Supported
- TrustZone Architecture for Enhanced OS Security

## 14.3 Product Dependencies

### 14.3.1 Power Management

The L2 Cache Controller is continuously clocked by the Processor Clock. The Power Management Controller has no effect on the behavior of the L2 Cache Controller.

## 14.4 Functional Description

The addition of an on-chip secondary cache, also referred to as a Level 2 or L2 cache, is a recognized method of improving the performance of ARM-based systems when significant memory traffic is generated by the processor. By definition a secondary cache assumes the presence of a Level 1 or primary cache, closely coupled or internal to the processor. Memory access is fastest to L1 cache, followed closely by L2 cache. Memory access is typically significantly slower with L3 main memory.

The cache controller is a unified, physically addressed, physically tagged cache with up to 8 ways. The user can lock the replacement algorithm on a way basis, enabling the associativity to be reduced from 8-way down to 1-way (directly mapped).

The cache controller does not have snooping hardware to maintain coherency between caches, so the user has to maintain coherency by software.

### 14.4.1 Double Linefill Issuing

The L2CC cache line length is 32-byte. Therefore, by default, on each L2 cache miss, the L2CC issues 32-byte linefills, 4 x 64-bit read bursts, to the L3 memory system. The L2CC can issue 64-byte linefills, 8 x 64-bit read bursts, on an L2 cache miss. When the L2CC is waiting for the data from L3, it performs a lookup on the second cache line targeted by the 64-byte linefill. If it misses, data corresponding to the second cache line are allocated to the L2 cache. If it hits, data corresponding to the second cache line are discarded.

The user can control this feature using the DLEN, DLFWRDIS and DLEN bits of the L2CC Prefetch Control Register. The IDLEN and DLFWRDIS bits are only used if the user sets the DLEN bit HIGH. The

table below shows the behavior of the L2CC master ports, depending on the configuration chosen by the user.

**Table 14-1. L2CC Master Port Behavior**

| Bit 30 DLEN | Bit 27 DLFWRDIS | Bit 23 IDLEN | Original Read Address from L1 | Read Address to L3 | AXI Burst Type | AXI Burst Length | Targeted Cache Lines |
|---|---|---|---|---|---|---|---|
| 0 | 0 or 1 | 0 or 1 | 0x00 | 0x00 | WRAP | 0x3, 4x64-bit | 0x00 |
| 0 | 0 or 1 | 0 or 1 | 0x20 | 0x20 | WRAP | 0x3, 4x64-bit | 0x20 |
| 1 | 0 or 1 | 0 | 0x00 | 0x00 | WRAP | 0x7, 8x64-bit | 0x00 and 0x20 |
| 1 | 1 | 0 | 0x08 or 0x10 or 0x18 | 0x08 | WRAP | 0x3, 4x64-bit | 0x00 |
| 1 | 0 | 0 | 0x08 or 0x10 or 0x18 | 0x00 | WRAP | 0x7, 8x64-bit | 0x00 and 0x20 |
| 1 | 0 or 1 | 0 | 0x20 | 0x20 | WRAP | 0x7, 8x64-bit | 0x00 and 0x20 |
| 1 | 1 | 0 | 0x28 or 0x30 or 0x38 | 0x28 | WRAP | 0x3, 4x64-bit | 0x20 |
| 1 | 0 | 0 | 0x28 or 0x30 or 0x38 | 0x20 | WRAP | 0x7, 8x64-bit | 0x00 and 0x20 |
| 1 | 0 or 1 | 1 | 0x00 | 0x00 | INCR or WRAP | 0x7, 8x64-bit | 0x00 and 0x20 |
| 1 | 1 | 1 | 0x08 or 0x10 or 0x18 | 0x08 | WRAP | 0x3, 4x64-bit | 0x00 |
| 1 | 0 | 1 | 0x08 or 0x10 or 0x18 | 0x00 | INCR or WRAP | 0x7, 8x64-bit | 0x00 and 0x20 |
| 1 | 0 or 1 | 1 | 0x20 | 0x20 | INCR | 0x7, 8x64-bit | 0x20 and 0x40 |
| 1 | 1 | 1 | 0x28 or 0x30 or 0x38 | 0x28 | WRAP | 0x3, 4x64-bit | 0x20 |
| 1 | 0 | 1 | 0x28 or 0x30 or 0x38 | 0x20 | INCR | 0x7, 8x64-bit | 0x20 and 0x40 |

**Note:**

1. Double linefills are not issued for prefetch reads if exclusive cache configuration is enabled.
2. Double linefills are not launched when crossing a 4-Kbyte boundary.
3. Double linefills only occur if a WRAP4 or an INCR4 64-bit transaction is received on the slave ports. This transaction is most commonly seen as a result of a cache linefill in a master, but can be produced by a master when accessing memory marked as inner non-cacheable.

## 14.5 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | L2CC_IDR | 7:0 | ID[7:0] | | | | | | | |
| | | 15:8 | ID[15:8] | | | | | | | |
| | | 23:16 | ID[23:16] | | | | | | | |
| | | 31:24 | ID[31:24] | | | | | | | |
| 0x04 | L2CC_TYPR | 7:0 | | IL2ASS | | | | | | |
| | | 15:8 | | | | | | IL2WSIZE[2:0] | | |
| | | 23:16 | | DL2WSIZE[2:0] | | | | DL2ASS | | |
| | | 31:24 | | | | | | | | |
| 0x08 ... 0xFF | Reserved | | | | | | | | | |
| 0x0100 | L2CC_CR | 7:0 | | | | | | | | L2CEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0104 | L2CC_ACR | 7:0 | | | | | | | | |
| | | 15:8 | | | SAIE | EXCC | SBDLE | HPSO | | |
| | | 23:16 | FWA[0:0] | SAOEN | PEN | EMBEN | WAYSIZE[2:0] | | | ASS |
| | | 31:24 | | | IPEN | DPEN | NSIAC | NSLEN | CRPOL | FWA[1:1] |
| 0x0108 | L2CC_TRCR | 7:0 | | TRDLAT[2:0] | | | | TSETLAT[2:0] | | |
| | | 15:8 | | | | | | TWRLAT[2:0] | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x010C | L2CC_DRCR | 7:0 | | DRDLAT[2:0] | | | | DSETLAT[2:0] | | |
| | | 15:8 | | | | | | DWRLAT[2:0] | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0110 ... 0x01FF | Reserved | | | | | | | | | |
| 0x0200 | L2CC_ECR | 7:0 | | | | | | EVC1RST | EVC0RST | EVCEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0204 | L2CC_ECFGR1 | 7:0 | | | ESRC[3:0] | | | | EIGEN[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0208 | L2CC_ECFGR0 | 7:0 | | | ESRC[3:0] | | | | EIGEN[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x020C | L2CC_EVR1 | 7:0 | VALUE[7:0] | | | | | | | |
| | | 15:8 | VALUE[15:8] | | | | | | | |
| | | 23:16 | VALUE[23:16] | | | | | | | |
| | | 31:24 | VALUE[31:24] | | | | | | | |
| 0x0210 | L2CC_EVR0 | 7:0 | VALUE[7:0] | | | | | | | |
| | | 15:8 | VALUE[15:8] | | | | | | | |
| | | 23:16 | VALUE[23:16] | | | | | | | |
| | | 31:24 | VALUE[31:24] | | | | | | | |
| 0x0214 | L2CC_IMR | 7:0 | SLVERR | ERRRD | ERRRT | ERRWD | ERRWT | PARRD | PARRT | ECNTR |
| | | 15:8 | | | | | | | | DECERR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0218 | L2CC_MISR | 7:0 | SLVERR | ERRRD | ERRRT | ERRWD | ERRWT | PARRD | PARRT | ECNTR |
| | | 15:8 | | | | | | | | DECERR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x021C | L2CC_RISR | 7:0 | SLVERR | ERRRD | ERRRT | ERRWD | ERRWT | PARRD | PARRT | ECNTR |
| | | 15:8 | | | | | | | | DECERR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0220 | L2CC_ICR | 7:0 | SLVERR | ERRRD | ERRRT | ERRWD | ERRWT | PARRD | PARRT | ECNTR |
| | | 15:8 | | | | | | | | DECERR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0224 ... 0x072F | Reserved | | | | | | | | | |
| 0x0730 | L2CC_CSR | 7:0 | | | | | | | | C |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0734 ... 0x076F | Reserved | | | | | | | | | |
| 0x0770 | L2CC_IPALR | 7:0 | IDX[2:0] | | | | | | | C |
| | | 15:8 | TAG[1:0] | | IDX[8:3] | | | | | |
| | | 23:16 | TAG[9:2] | | | | | | | |
| | | 31:24 | TAG[17:10] | | | | | | | |
| 0x0774 ... 0x077B | Reserved | | | | | | | | | |
| 0x077C | L2CC_IWR | 7:0 | WAY7 | WAY6 | WAY5 | WAY4 | WAY3 | WAY2 | WAY1 | WAY0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0780 ... 0x07AF | Reserved | | | | | | | | | |
| 0x07B0 | L2CC_CPALR | 7:0 | IDX[2:0] | | | | | | | C |
| | | 15:8 | TAG[1:0] | | IDX[8:3] | | | | | |
| | | 23:16 | TAG[9:2] | | | | | | | |
| | | 31:24 | TAG[17:10] | | | | | | | |
| 0x07B4 ... 0x07B7 | Reserved | | | | | | | | | |
| 0x07B8 | L2CC_CIR | 7:0 | IDX[2:0] | | | | | | | C |
| | | 15:8 | | | IDX[8:3] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | WAY[2:0] | | | | | | |
| 0x07BC | L2CC_CWR | 7:0 | WAY7 | WAY6 | WAY5 | WAY4 | WAY3 | WAY2 | WAY1 | WAY0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x07C0 ... 0x07EF | Reserved | | | | | | | | | |
| 0x07F0 | L2CC_CIPALR | 7:0 | IDX[2:0] | | | | | | | C |
| | | 15:8 | TAG[1:0] | | IDX[8:3] | | | | | |
| | | 23:16 | TAG[9:2] | | | | | | | |
| | | 31:24 | TAG[17:10] | | | | | | | |
| 0x07F4 ... 0x07F7 | Reserved | | | | | | | | | |
| 0x07F8 | L2CC_CIIR | 7:0 | IDX[2:0] | | | | | | | C |
| | | 15:8 | | | IDX[8:3] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | WAY[2:0] | | | | | | |
| 0x07FC | L2CC_CIWR | 7:0 | WAY7 | WAY6 | WAY5 | WAY4 | WAY3 | WAY2 | WAY1 | WAY0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0800 ... 0x08FF | Reserved | | | | | | | | | |
| 0x0900 | L2CC_DLKR | 7:0 | DLK7 | DLK6 | DLK5 | DLK4 | DLK3 | DLK2 | DLK1 | DLK0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0904 | L2CC_ILKR | 7:0 | ILK7 | ILK6 | ILK5 | ILK4 | ILK3 | ILK2 | ILK1 | ILK0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0908 ... 0x0F3F | Reserved | | | | | | | | | |
| 0x0F40 | L2CC_DCR | 7:0 | | | | | | SPNIDEN | DWB | DCL |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0F44 ... 0x0F5F | Reserved | | | | | | | | | |
| 0x0F60 | L2CC_PCR | 7:0 | | | | OFFSET[4:0] | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | IDLEN | | NSIDEN | | | | | |
| | | 31:24 | | DLEN | INSPEN | DATPEN | DLFWRDIS | | | PDEN |
| 0x0F64 ... 0x0F7F | Reserved | | | | | | | | | |
| 0x0F80 | L2CC_POWCR | 7:0 | | | | | | | DCKGATEN | STBYEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 14.5.1 L2CC Cache ID Register

**Name:** L2CC_IDR
**Offset:** 0x000
**Reset:** 0x410000C9
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ID[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ID[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ID[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ID[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |

**Bits 31:0 – ID[31:0]** Cache Controller ID
The cache ID is 0x410000C9.

### 14.5.2 L2CC Type Register

**Name:** L2CC_TYPR
**Offset:** 0x004
**Reset:** 0x00100100
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | DL2WSIZE[2:0] | | | | DL2ASS | | |
| Access | | R | R | R | | R | | |
| Reset | | 0 | 0 | 1 | | 0 | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | IL2WSIZE[2:0] | | |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | IL2ASS | | | | | | |
| Access | | R | | | | | | |
| Reset | | 0 | | | | | | |

**Bits 22:20 – DL2WSIZE[2:0]** Data L2 Cache Way Size
The value is read from the field WAYSIZE in Auxiliary Control Register, should be 0x1.

**Bit 18 – DL2ASS** Data L2 Cache Associativity
The value is read from the field ASS in Auxiliary Control Register, should be 0.

**Bits 10:8 – IL2WSIZE[2:0]** Instruction L2 Cache Way Size
The value is read from the field WAYSIZE in Auxiliary Control Register, should be 0x1.

**Bit 6 – IL2ASS** Instruction L2 Cache Associativity
The value is read from the field ASS in Auxiliary Control Register, should be 0.

### 14.5.3 L2CC Control Register

**Name:** L2CC_CR
**Offset:** 0x100
**Reset:** 0x00000000
**Property:** Read/Write in Secure mode, Read-only in Non-secure mode

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | L2CEN |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

**Bit 0 – L2CEN** L2 Cache Enable

| Value | Description |
|---|---|
| 0 | L2 Cache is disabled. This is the default value. |
| 1 | L2 Cache is enabled. |

### 14.5.4 L2CC Auxiliary Control Register

**Name:** L2CC_ACR
**Offset:** 0x104
**Reset:** 0x02020000
**Property:** Read/Write in Secure mode, Read-only in Non-secure mode

The L2 Cache Controller (L2CC) must be disabled in the L2CC Control Register prior to any write access to this register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | IPEN | DPEN | NSIAC | NSLEN | CRPOL | FWA[1:1] |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | 1 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FWA[0:0] | SAOEN | PEN | EMBEN | WAYSIZE[2:0] | | | ASS |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | SAIE | EXCC | SBDLE | HPSO | | |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 29 – IPEN** Instruction Prefetch Enable

| Value | Description |
|---|---|
| 0 | Instruction prefetching is disabled. This is the default value. |
| 1 | Instruction prefetching is enabled. |

**Bit 28 – DPEN** Data Prefetch Enable

| Value | Description |
|---|---|
| 0 | Data prefetching is disabled. This is the default value. |
| 1 | Data prefetching is enabled. |

**Bit 27 – NSIAC** Non-Secure Interrupt Access Control

| Value | Description |
|---|---|
| 0 | Interrupt Clear Register and Interrupt Mask Register can only be modified or read with secure accesses. This is the default value. |
| 1 | Interrupt Clear Register and Interrupt Mask Register can be modified or read with secure or non-secure accesses. |

**Bit 26 – NSLEN** Non-Secure Lockdown Enable

| Value | Description |
|---|---|
| 0 | Lockdown registers cannot be modified using non-secure accesses. This is the default value. |
| 1 | Non-secure accesses can write to the lockdown registers. |

**Bit 25 – CRPOL** Cache Replacement Policy

| Value | Description |
|---|---|
| 0 | Pseudo-random replacement using the LFSR algorithm. |
| 1 | Round-robin replacement. This is always the default value. |

**Bits 24:23 – FWA[1:0]** Force Write Allocate

| Value | Description |
|---|---|
| 0 | The L2 Cache controller uses AWCACHE attributes for WA. This is the default value. |
| 1 | User forces no allocate, WA bit must be set to 0. |
| 2 | User overrides AWCACHE attributes, WA bit must be set to 1. All cacheable write misses become write allocated. |
| 3 | The write allocation is internally mapped to 00. |

**Bit 22 – SAOEN** Shared Attribute Override Enable

| Value | Description |
|---|---|
| 0 | Treats shared accesses. This is the default value. |
| 1 | Shared attribute is internally ignored. |

**Bit 21 – PEN** Parity Enable

| Value | Description |
|---|---|
| 0 | Disabled. This is the default value. |
| 1 | Enabled. |

**Bit 20 – EMBEN** Event Monitor Bus Enable

| Value | Description |
|---|---|
| 0 | Disabled. This is the default value. |
| 1 | Enabled. |

**Bits 19:17 – WAYSIZE[2:0]** Way Size

| Value | Name | Description |
|---|---|---|
| 0x0 | RESERVED | Reserved |
| 0x1 | 16KB_WAY | 16-Kbyte way set associative |
| 0x2 | RESERVED | Reserved |
| 0x3 | RESERVED | Reserved |
| 0x4 | RESERVED | Reserved |
| 0x5 | RESERVED | Reserved |
| 0x6 | RESERVED | Reserved |
| 0x7 | RESERVED | Reserved |

**Bit 16 – ASS** Associativity

| Value | Description |
|---|---|
| 0 | 8-way.This is the default value. |
| 1 | Reserved. |

**Bit 13 – SAIE** Shared Attribute Invalidate Enable

| Value | Description |
|---|---|
| 0 | Shared invalidate behavior is disabled. This is the default value. |
| 1 | Shared invalidate behavior is enabled if the Shared Attribute Override Enable bit is not set.<br>Shared invalidate behavior is enabled if both:<br>• Shareable Attribute Invalidate Enable bit is set in the Auxiliary Control Register, bit[13]<br>• Shared Attribute Override Enable bit is not set in the Auxiliary Control Register, bit[22] |

**Bit 12 – EXCC** Exclusive Cache Configuration

| Value | Description |
|---|---|
| 0 | Disabled. This is the default value. |
| 1 | Enabled. |

**Bit 11 – SBDLE** Store Buffer Device Limitation Enable

| Value | Description |
|---|---|
| 0 | Store buffer device limitation is disabled. Device writes can take all slots in the store buffer. This is the default value. |
| 1 | Store buffer device limitation is enabled. |

**Bit 10 – HPSO** High Priority for SO and Dev Reads Enable

| Value | Description |
|---|---|
| 0 | Strongly Ordered and Device reads have lower priority than cacheable accesses when arbitrated in the L2CC master ports. This is the default value. |
| 1 | Strongly Ordered and Device reads get the highest priority when arbitrated in the L2CC master ports. |

### 14.5.5 L2CC Tag RAM Latency Control Register

**Name:** L2CC_TRCR
**Offset:** 0x108
**Reset:** 0x00000111
**Property:** Read/Write in Secure mode, Read-only in Non-secure mode

The L2 Cache Controller (L2CC) must be disabled in the L2CC Control Register prior to any write access to this register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | TWRLAT[2:0] | | |
| Reset | | | | | | 0 | 0 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | TRDLAT[2:0] | | | | TSETLAT[2:0] | | |
| Reset | | 0 | 0 | 1 | | 0 | 0 | 1 |

**Bits 10:8 – TWRLAT[2:0]** Write Access Latency
Latency to Tag RAM is the programmed value + 1.

Default value is 0.

**Bits 6:4 – TRDLAT[2:0]** Read Access Latency

**Bits 2:0 – TSETLAT[2:0]** Setup Latency

### 14.5.6 L2CC Data RAM Latency Control Register

**Name:** L2CC_DRCR
**Offset:** 0x10C
**Reset:** 0x00000111
**Property:** Read/Write in Secure mode, Read-only in Non-secure mode

The L2 Cache Controller (L2CC) must be disabled in the L2CC Control Register prior to any write access to this register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | DWRLAT[2:0] | | |
| Reset | | | | | | 0 | 0 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | DRDLAT[2:0] | | | | DSETLAT[2:0] | | |
| Reset | | 0 | 0 | 1 | | 0 | 0 | 1 |

**Bits 10:8 – DWRLAT[2:0]** Write Access Latency
Latency to Data RAM is the programmed value + 1.

Default value is 0.

**Bits 6:4 – DRDLAT[2:0]** Read Access Latency

**Bits 2:0 – DSETLAT[2:0]** Setup Latency

### 14.5.7 L2CC Event Counter Control Register

**Name:** L2CC_ECR
**Offset:** 0x200
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | EVC1RST | EVC0RST | EVCEN |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – EVC1RST** Event Counter 1 Reset

| Value | Description |
|---|---|
| 0 | No effect, always read as zero. |
| 1 | Resets Counter 1. |

**Bit 1 – EVC0RST** Event Counter 0 Reset

| Value | Description |
|---|---|
| 0 | No effect, always read as zero. |
| 1 | Resets Counter 0. |

**Bit 0 – EVCEN** Event Counter Enable

| Value | Description |
|---|---|
| 0 | Disables Event Counter. This is the default value. |
| 1 | Enables Event Counter. |

### 14.5.8 L2CC Event Counter 1 Configuration Register

**Name:** L2CC_ECFGR1
**Offset:** 0x204
**Reset:** 0x02020000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | ESRC[3:0] | | | | EIGEN[1:0] | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:2 – ESRC[3:0]** Event Counter Source

| Value | Name | Description |
|---|---|---|
| 0x0 | CNT_DIS | Counter Disabled |
| 0x1 | SRC_CO | Source is CO |
| 0x2 | SRC_DRHIT | Source is DRHIT |
| 0x3 | SRC_DRREQ | Source is DRREQ |
| 0x4 | SRC_DWHIT | Source is DWHIT |
| 0x5 | SRC_DWREQ | Source is DWREQ |
| 0x6 | SRC_DWTREQ | Source is DWTREQ |
| 0x7 | SRC_IRHIT | Source is IRHIT |
| 0x8 | SRC_IRREQ | Source is IRREQ |
| 0x9 | SRC_WA | Source is WA |
| 0xa | SRC_IPFALLOC | Source is IPFALLOC |
| 0xb | SRC_EPFHIT | Source is EPFHIT |
| 0xc | SRC_EPFALLOC | Source is EPFALLOC |
| 0xd | SRC_SRRCVD | Source is SRRCVD |
| 0xe | SRC_SRCONF | Source is SRCONF |
| 0xf | SRC_EPFRCVD | Source is EPFRCVD |

**Bits 1:0 – EIGEN[1:0]** Event Counter Interrupt Generation

| Value | Name | Description |
|---|---|---|
| 0x0 | INT_DIS | Disables (default) |
| 0x1 | INT_EN_INCR | Enables with Increment condition |
| 0x2 | INT_EN_OVER | Enables with Overflow condition |
| 0x3 | INT_GEN_DIS | Disables Interrupt generation |

### 14.5.9 L2CC Event Counter 0 Configuration Register

**Name:** L2CC_ECFGR0
**Offset:** 0x208
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | ESRC[3:0] | | | | EIGEN[1:0] | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:2 – ESRC[3:0]** Event Counter Source

| Value | Name | Description |
|---|---|---|
| 0x0 | CNT_DIS | Counter Disabled |
| 0x1 | SRC_CO | Source is CO |
| 0x2 | SRC_DRHIT | Source is DRHIT |
| 0x3 | SRC_DRREQ | Source is DRREQ |
| 0x4 | SRC_DWHIT | Source is DWHIT |
| 0x5 | SRC_DWREQ | Source is DWREQ |
| 0x6 | SRC_DWTREQ | Source is DWTREQ |
| 0x7 | SRC_IRHIT | Source is IRHIT |
| 0x8 | SRC_IRREQ | Source is IRREQ |
| 0x9 | SRC_WA | Source is WA |
| 0xa | SRC_IPFALLOC | Source is IPFALLOC |
| 0xb | SRC_EPFHIT | Source is EPFHIT |
| 0xc | SRC_EPFALLOC | Source is EPFALLOC |
| 0xd | SRC_SRRCVD | Source is SRRCVD |
| 0xe | SRC_SRCONF | Source is SRCONF |
| 0xf | SRC_EPFRCVD | Source is EPFRCVD |

**Bits 1:0 – EIGEN[1:0]** Event Counter Interrupt Generation

| Value | Name | Description |
|---|---|---|
| 0x0 | INT_DIS | Disables (default) |
| 0x1 | INT_EN_INCR | Enables with Increment condition |
| 0x2 | INT_EN_OVER | Enables with Overflow condition |
| 0x3 | INT_GEN_DIS | Disables Interrupt generation |

### 14.5.10 L2CC Event Counter 1 Value Register

**Name:** L2CC_EVR1
**Offset:** 0x20C
**Reset:** 0x00000000
**Property:** Read/Write

Counter 1 must be disabled in the L2CC Event Counter 1 Configuration Register prior to any write access to this register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | VALUE[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | VALUE[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | VALUE[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | VALUE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – VALUE[31:0]** Event Counter Value
Value returns the number of instance of the selected event.

If a counter reaches its maximum value, it remains saturated at that value until it is reset.

### 14.5.11 L2CC Event Counter 0 Value Register

**Name:** L2CC_EVR0
**Offset:** 0x210
**Reset:** 0x00000000
**Property:** Read/Write

Counter 0 must be disabled in the L2CC Event Counter 0 Configuration Register prior to any write access to this register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | VALUE[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | VALUE[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | VALUE[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | VALUE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – VALUE[31:0]** Event Counter Value
Value returns the number of instance of the selected event.

If a counter reaches its maximum value, it remains saturated at that value until it is reset.

### 14.5.12 L2CC Interrupt Mask Register

**Name:** L2CC_IMR
**Offset:** 0x214
**Reset:** 0x00000000
**Property:** Programmable in Auxiliary Control register.

The following configuration values are valid for all listed bit names of this register:

0: Masked. This is the default value.

1: Enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 8 – DECERR** DECERR from L3 Memory

**Bit 7 – SLVERR** SLVERR from L3 Memory

**Bit 6 – ERRRD** Error on L2 Data RAM, Read

**Bit 5 – ERRRT** Error on L2 Tag RAM, Read

**Bit 4 – ERRWD** Error on L2 Data RAM, Write

**Bit 3 – ERRWT** Error on L2 Tag RAM, Write

**Bit 2 – PARRD** Parity Error on L2 Data RAM, Read

**Bit 1 – PARRT** Parity Error on L2 Tag RAM, Read

**Bit 0 – ECNTR** Event Counter 1/0 Overflow Increment

### 14.5.13 L2CC Masked Interrupt Status Register

**Name:** L2CC_MISR
**Offset:** 0x218
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: No interrupt has been generated or the interrupt is masked.

1: The input lines have triggered an interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DECERR |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SLVERR | ERRRD | ERRRT | ERRWD | ERRWT | PARRD | PARRT | ECNTR |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 8 – DECERR** DECERR from L3 memory

**Bit 7 – SLVERR** SLVERR from L3 memory

**Bit 6 – ERRRD** Error on L2 Data RAM, Read

**Bit 5 – ERRRT** Error on L2 Tag RAM, Read

**Bit 4 – ERRWD** Error on L2 Data RAM, Write

**Bit 3 – ERRWT** Error on L2 Tag RAM, Write

**Bit 2 – PARRD** Parity Error on L2 Data RAM, Read

**Bit 1 – PARRT** Parity Error on L2 Tag RAM, Read

**Bit 0 – ECNTR** Event Counter 1/0 Overflow Increment

### 14.5.14 L2CC Raw Interrupt Status Register

**Name:** L2CC_RISR
**Offset:** 0x21C
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: No interrupt has been generated.

1: The input lines have triggered an interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DECERR |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SLVERR | ERRRD | ERRRT | ERRWD | ERRWT | PARRD | PARRT | ECNTR |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 8 – DECERR** DECERR from L3 memory

**Bit 7 – SLVERR** SLVERR from L3 memory

**Bit 6 – ERRRD** Error on L2 Data RAM, Read

**Bit 5 – ERRRT** Error on L2 Tag RAM, Read

**Bit 4 – ERRWD** Error on L2 Data RAM, Write

**Bit 3 – ERRWT** Error on L2 Tag RAM, Write

**Bit 2 – PARRD** Parity Error on L2 Data RAM, Read

**Bit 1 – PARRT** Parity Error on L2 Tag RAM, Read

**Bit 0 – ECNTR** Event Counter 1/0 Overflow Increment

### 14.5.15 L2CC Interrupt Clear Register

**Name:** L2CC_ICR
**Offset:** 0x220
**Reset:** 0x00000000
**Property:** Programmable in Auxiliary Control register.

The following configuration values are valid for all listed bit names of this register:

0: No effect. Read returns zero.

1: Clears the corresponding bit in the Raw Interrupt Status Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DECERR |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SLVERR | ERRRD | ERRRT | ERRWD | ERRWT | PARRD | PARRT | ECNTR |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 8 – DECERR** DECERR from L3 memory

**Bit 7 – SLVERR** SLVERR from L3 memory

**Bit 6 – ERRRD** Error on L2 Data RAM, Read

**Bit 5 – ERRRT** Error on L2 Tag RAM, Read

**Bit 4 – ERRWD** Error on L2 Data RAM, Write

**Bit 3 – ERRWT** Error on L2 Tag RAM, Write

**Bit 2 – PARRD** Parity Error on L2 Data RAM, Read

**Bit 1 – PARRT** Parity Error on L2 Tag RAM, Read

**Bit 0 – ECNTR** Event Counter 1/0 Overflow Increment

### 14.5.16 L2CC Cache Synchronization Register

**Name:** L2CC_CSR
**Offset:** 0x730
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | C |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – C** Cache Synchronization Status

| Value | Description |
|---|---|
| 0 | No background operation is in progress. When written, must be zero. |
| 1 | A background operation is in progress. |

### 14.5.17 L2CC Invalidate Physical Address Line Register

**Name:** L2CC_IPALR
**Offset:** 0x770
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TAG[17:10] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TAG[9:2] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TAG[1:0] | | IDX[8:3] | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IDX[2:0] | | | | | | | C |
| Access | R/W | R/W | R/W | | | | | R/W |
| Reset | 0 | 0 | 0 | | | | | 0 |

**Bits 31:14 – TAG[17:0]** Tag Number

**Bits 13:5 – IDX[8:0]** Index Number

**Bit 0 – C** Cache Synchronization Status

| Value | Description |
|---|---|
| 0 | No background operation is in progress. When written, must be zero. |
| 1 | A background operation is in progress. |

### 14.5.18 L2CC Invalidate Way Register

**Name:** L2CC_IWR
**Offset:** 0x77C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WAY7 | WAY6 | WAY5 | WAY4 | WAY3 | WAY2 | WAY1 | WAY0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – WAYx** Invalidate Way Number x

| Value | Description |
|---|---|
| 0 | The corresponding way is totally invalidated. |
| 1 | Invalidates the way. This bit is read as ‘1’ as long as invalidation of the way is in progress. |

### 14.5.19 L2CC Clean Physical Address Line Register

**Name:** L2CC_CPALR
**Offset:** 0x7B0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TAG[17:10] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TAG[9:2] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TAG[1:0] | | IDX[8:3] | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IDX[2:0] | | | | | | | C |
| Access | R/W | R/W | R/W | | | | | R/W |
| Reset | 0 | 0 | 0 | | | | | 0 |

**Bits 31:14 – TAG[17:0]** Tag Number

**Bits 13:5 – IDX[8:0]** Index Number

**Bit 0 – C** Cache Synchronization Status

| Value | Description |
|---|---|
| 0 | No background operation is in progress. When written, must be zero. |
| 1 | A background operation is in progress. |

### 14.5.20 L2CC Clean Index Register

**Name:** L2CC_CIR
**Offset:** 0x7B8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | WAY[2:0] | | | | | | |
| Access | | R/W | R/W | R/W | | | | |
| Reset | | 0 | 0 | 0 | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | IDX[8:3] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IDX[2:0] | | | | | | | C |
| Access | R/W | R/W | R/W | | | | | R/W |
| Reset | 0 | 0 | 0 | | | | | 0 |

**Bits 30:28 – WAY[2:0]** Way Number

**Bits 13:5 – IDX[8:0]** Index Number

**Bit 0 – C** Cache Synchronization Status

| Value | Description |
|---|---|
| 0 | No background operation is in progress. When written, must be zero. |
| 1 | A background operation is in progress. |

### 14.5.21 L2CC Clean Way Register

**Name:** L2CC_CWR
**Offset:** 0x7BC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WAY7 | WAY6 | WAY5 | WAY4 | WAY3 | WAY2 | WAY1 | WAY0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – WAYx** Clean Way Number x

| Value | Description |
|---|---|
| 0 | The corresponding way is totally cleaned. |
| 1 | Cleans the way. This bit is read as ‘1’ as long as cleaning of the way is in progress. |

### 14.5.22 L2CC Clean Invalidate Physical Address Line Register

**Name:** L2CC_CIPALR
**Offset:** 0x7F0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TAG[17:10] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TAG[9:2] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TAG[1:0] | | IDX[8:3] | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IDX[2:0] | | | | | | | C |
| Access | R/W | R/W | R/W | | | | | R/W |
| Reset | 0 | 0 | 0 | | | | | 0 |

**Bits 31:14 – TAG[17:0]** Tag Number

**Bits 13:5 – IDX[8:0]** Index Number

**Bit 0 – C** Cache Synchronization Status

| Value | Description |
|---|---|
| 0 | No background operation is in progress. When written, must be zero. |
| 1 | A background operation is in progress. |

### 14.5.23 L2CC Clean Invalidate Index Register

**Name:** L2CC_CIIR
**Offset:** 0x7F8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | WAY[2:0] | | | | | | |
| Access | | R/W | R/W | R/W | | | | |
| Reset | | 0 | 0 | 0 | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | IDX[8:3] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IDX[2:0] | | | | | | | C |
| Access | R/W | R/W | R/W | | | | | R/W |
| Reset | 0 | 0 | 0 | | | | | 0 |

**Bits 30:28 – WAY[2:0]** Way Number

**Bits 13:5 – IDX[8:0]** Index Number

**Bit 0 – C** Cache Synchronization Status

| Value | Description |
|---|---|
| 0 | No background operation is in progress. When written, must be zero. |
| 1 | A background operation is in progress. |

### 14.5.24 L2CC Clean Invalidate Way Register

**Name:** L2CC_CIWR
**Offset:** 0x7FC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WAY7 | WAY6 | WAY5 | WAY4 | WAY3 | WAY2 | WAY1 | WAY0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – WAYx** Clean Invalidate Way Number x

| Value | Description |
|---|---|
| 0 | The corresponding way is totally invalidated and cleaned. |
| 1 | Invalidates and cleans the way. This bit is read as ‘1’ as long as invalidation and cleaning of the way is in progress. |

### 14.5.25 L2CC Data Lockdown Register

**Name:** L2CC_DLKR
**Offset:** 0x900
**Reset:** 0x00000000
**Property:** Programmable in Auxiliary Control register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DLK7 | DLK6 | DLK5 | DLK4 | DLK3 | DLK2 | DLK1 | DLK0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – DLKx** Data Lockdown in Way Number x

| Value | Description |
|---|---|
| 0 | Allocation can occur in the corresponding way. |
| 1 | There is no allocation in the corresponding way. |

### 14.5.26 L2CC Instruction Lockdown Register

**Name:** L2CC_ILKR
**Offset:** 0x904
**Reset:** 0x00000000
**Property:** Programmable in Auxiliary Control register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ILK7 | ILK6 | ILK5 | ILK4 | ILK3 | ILK2 | ILK1 | ILK0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – ILKx** Instruction Lockdown in Way Number x

| Value | Description |
|---|---|
| 0 | Allocation can occur in the corresponding way. |
| 1 | There is no allocation in the corresponding way. |

### 14.5.27 L2CC Debug Control Register

**Name:** L2CC_DCR
**Offset:** 0xF40
**Reset:** 0x00000000
**Property:** Read/Write in Secure mode, Read-only in Non-secure mode

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SPNIDEN | DWB | DCL |
| Access | | | | | | | | |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – SPNIDEN** SPNIDEN Value
Reads value of the SPNIDEN input.

**Bit 1 – DWB** Disable Write-back, Force Write-through

| Value | Description |
|---|---|
| 0 | Enables write-back behavior. This is the default value. |
| 1 | Forces write-through behavior. |

**Bit 0 – DCL** Disable Cache Linefill

| Value | Description |
|---|---|
| 0 | Enables cache linefills. This is the default value. |
| 1 | Disables cache linefills. |

### 14.5.28 L2CC Prefetch Control Register

**Name:** L2CC_PCR
**Offset:** 0xF60
**Reset:** 0x00000000
**Property:** Read/Write in Secure mode, Read-only in Non-secure mode

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | DLEN | INSPEN | DATPEN | DLFWRDIS | | | PDEN |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IDLEN | | NSIDEN | | | | | |
| Access | | | | | | | | |
| Reset | 0 | | 0 | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | OFFSET[4:0] | | | | |
| Access | | | | | | | | |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bit 30 – DLEN** Double Linefill Enable
See 14.4.1 Double Linefill Issuing for details on double linefill functionality.

| Value | Description |
|---|---|
| 0 | The L2CC always issues 4x64-bit read bursts to L3 on reads that miss in the L2 cache. This is the default value. |
| 1 | The L2CC issues 8x64-bit read bursts to L3 on reads that miss in the L2 cache. |

**Bit 29 – INSPEN** Instruction Prefetch Enable

| Value | Description |
|---|---|
| 0 | Instruction prefetching is disabled. This is the default value. |
| 1 | Instruction prefetching is enabled. |

**Bit 28 – DATPEN** Data Prefetch Enable

| Value | Description |
|---|---|
| 0 | Data prefetching is disabled. This is the default value. |
| 1 | Data prefetching is enabled. |

**Bit 27 – DLFWRDIS** Double Linefill on WRAP Read Disable

| Value | Description |
|---|---|
| 0 | Double linefill on WRAP read is enabled. This is the default value. |
| 1 | Double linefill on WRAP read is disabled.<br>Note: This bit can only be used if the DLEN bit is set HIGH. See 14.4.1 Double Linefill Issuing for details on double linefill functionality. |

**Bit 24 – PDEN** Prefetch Drop Enable

| Value | Description |
|---|---|
| 0 | The L2CC does not discard prefetch reads issued to L3. This is the default value. |
| 1 | The L2CC discards prefetch reads issued to L3 when there is a resource conflict with explicit reads. |

**Bit 23 – IDLEN** INCR Double Linefill Enable
This bit can only be used if the DLEN bit is set HIGH. See 14.4.1 Double Linefill Issuing for details on double linefill functionality.

| Value | Description |
|---|---|
| 0 | The L2CC does not issue INCR 8x64-bit read bursts to L3 on reads that miss in the L2 cache. This is the default value. |
| 1 | The L2CC can issue INCR 8x64-bit read bursts to L3 on reads that miss in the L2 cache. |

**Bit 21 – NSIDEN** Not Same ID on Exclusive Sequence Enable

| Value | Description |
|---|---|
| 0 | Read and write portions of a non-cacheable exclusive sequence have the same AXI ID when issued to L3. This is the default value. |
| 1 | Read and write portions of a non-cacheable exclusive sequence do not have the same AXI ID when issued to L3. |

**Bits 4:0 – OFFSET[4:0]** Prefetch Offset
Only use the Prefetch offset values of 0 to 7, 15, 23, and 31 for these bits. The L2CC does not support the other values.

### 14.5.29 L2CC Power Control Register

**Name:** L2CC_POWCR
**Offset:** 0xF80
**Reset:** 0x00000000
**Property:** Read/Write in Secure mode, Read-only in Non-secure mode

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | DCKGATEN | STBYEN |
| Access | | | | | | | | |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – DCKGATEN** Dynamic Clock Gating Enable

| Value | Description |
|---|---|
| 0 | Disabled. This is the default value. |
| 1 | Enabled. |

**Bit 0 – STBYEN** Standby Mode Enable

| Value | Description |
|---|---|
| 0 | Disabled. This is the default value. |
| 1 | Enabled. |

# 15. Debug and Test Features

## 15.1 Description

The device features a number of complementary debug and test capabilities.

A common JTAG/ICE (In-Circuit Emulator) port is used for standard debugging functions, such as downloading code and single-stepping through programs.

A 2-pin debug port Serial Wire Debug (SWD) replaces the 5-pin JTAG port and provides an easy and risk-free alternative to JTAG as the two signals, SWDIO and SWCLK, are overlaid on the TMS and TCK pins, allowing for bi-modal devices that provide the other JTAG signals. These extra JTAG pins can be switched to other uses when in SWD mode.

A set of dedicated debug and test input/output pins gives direct access to these capabilities from a PC-based test environment.

## 15.2 Embedded Characteristics

- Cortex-A5 In-circuit Emulator
  - Two real-time watchpoint units
  - Two independent registers: Debug Control Register and Debug Status Register
  - Test access port accessible through JTAG protocol
  - Debug communications channel
  - Serial wire debug
  - Trace
- Chip ID Register
- IEEE1149.1 JTAG Boundary-scan on All Digital Pins
- ETM, ETB: 8-Kbyte Embedded Trace Buffer

## 15.3 Debug and Test Block Diagrams

**Figure 15-1. Debug and Test General Block Diagram**

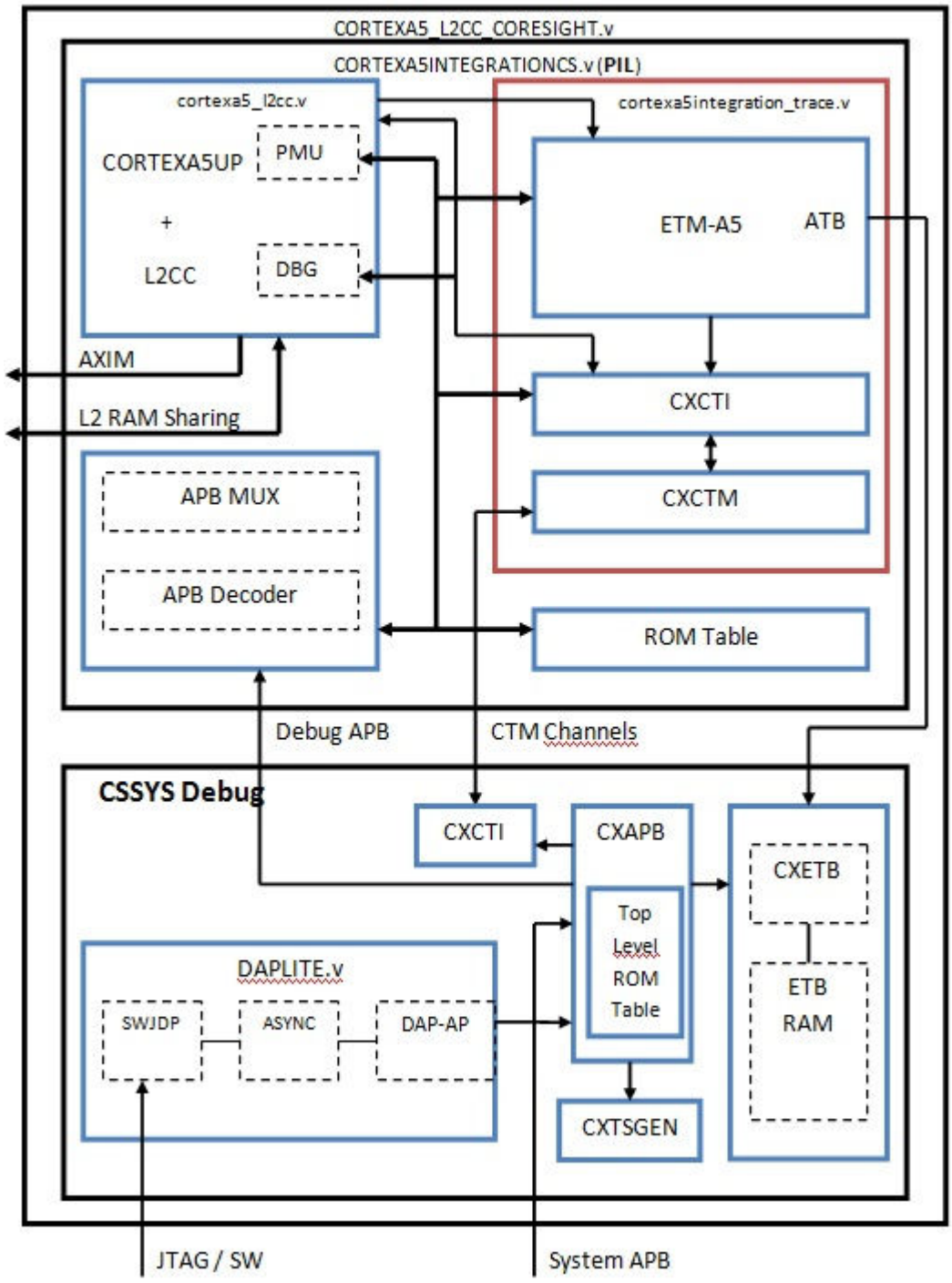
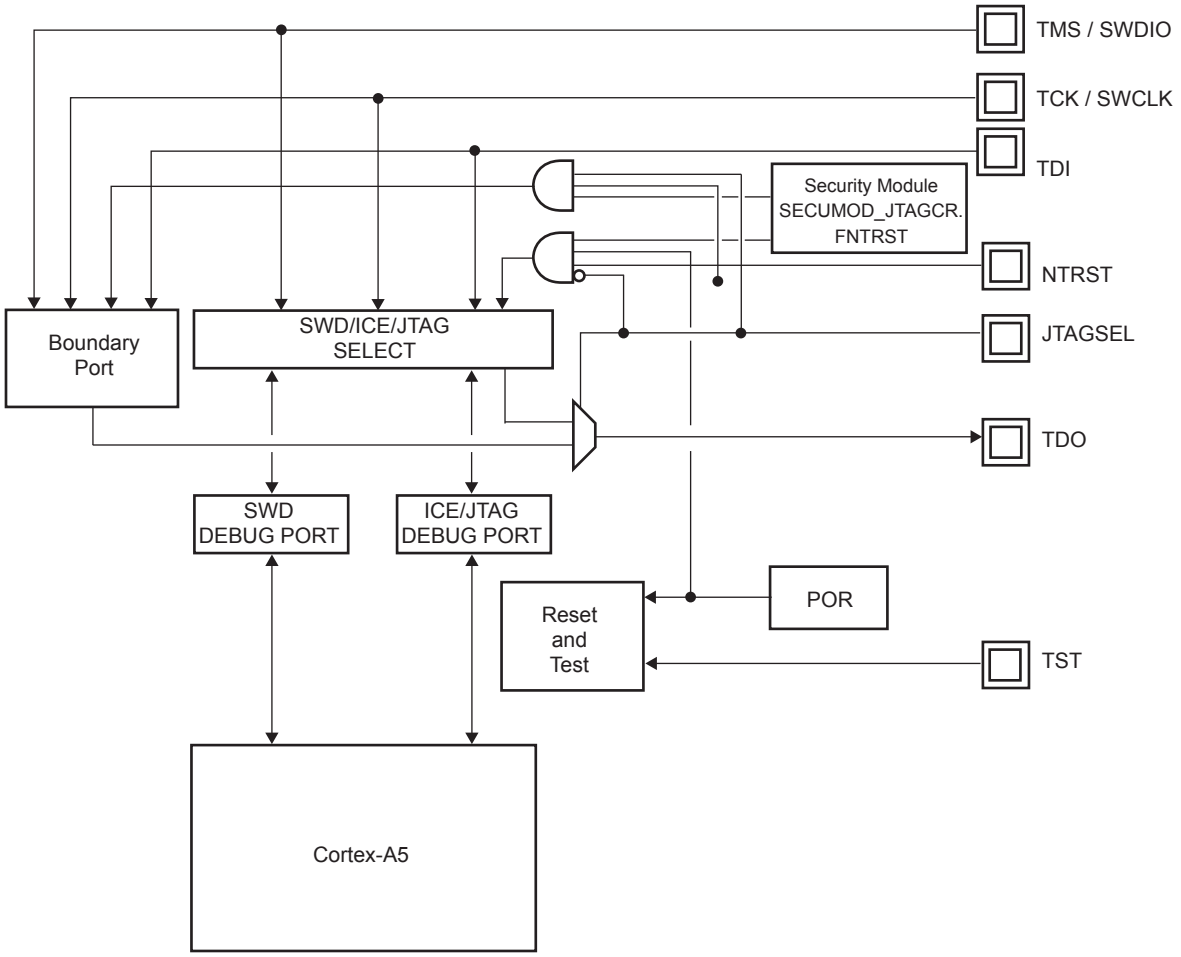


**Figure 15-2. Debug and Test Interface Block Diagram**

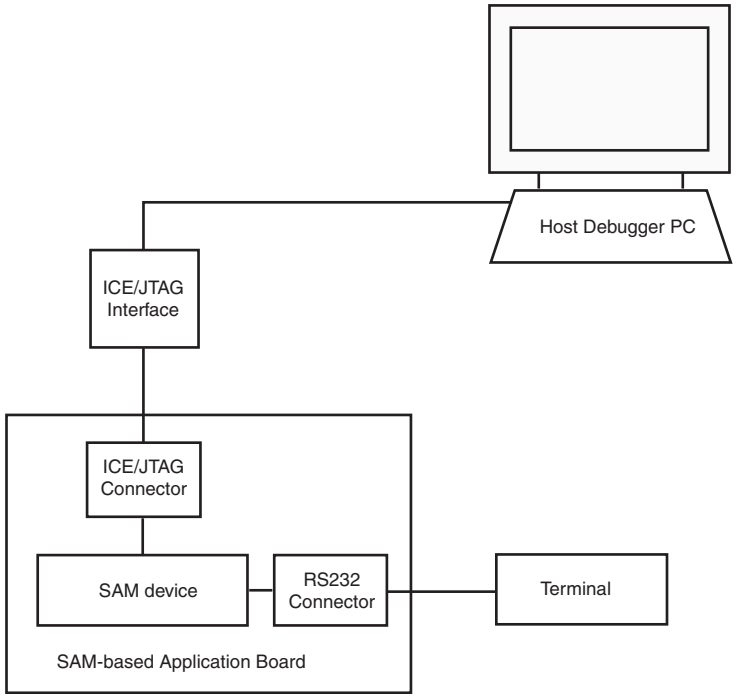



## 15.4 Application Examples

### 15.4.1 Debug Environment

The figure below shows a complete debug environment example. The ICE/JTAG interface is used for standard debugging functions, such as downloading code and single-stepping through the program. A software debugger running on a personal computer provides the user interface for configuring a Trace Port interface utilizing the ICE/JTAG interface.

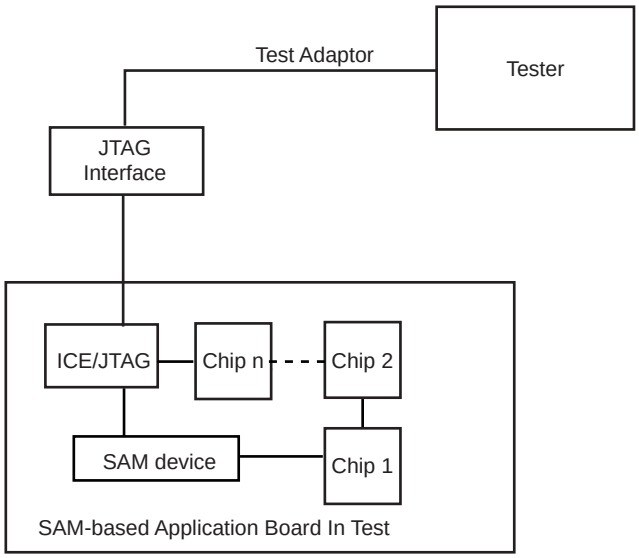
**Figure 15-3. Application Debug and Trace Environment Example**




### 15.4.2 Test Environment

The figure below shows a test environment example. Test vectors are sent and interpreted by the tester. In this example, the “board in test” is designed using a number of JTAG-compliant devices. These devices can be connected to form a single scan chain.

**Figure 15-4. Application Test Environment Example**



## 15.5 Debug and Test Pin Description

**Table 15-1. Debug and Test Pin List**

| Pin Name | Function | Type | Active Level |
|---|---|---|---|
| **Reset/Test** | | | |
| NRST | Microprocessor Reset | Input | Low |
| TST | Test Mode Select | Input | – |
| NTRST | Test Reset Signal | Input | – |
| **ICE and JTAG** | | | |
| TCK/SWCLK | Test Clock/Serial Wire Clock | Input | – |
| TDI | Test Data In | Input | – |
| TDO | Test Data Out | Output | – |
| TMS/SWDIO | Test Mode Select/Serial Wire Input/Output | I/O | – |
| JTAGSEL | JTAG Selection | Input | – |

## 15.6 Functional Description

### 15.6.1 Test Pin

One dedicated pin, TST, is used to define the device operating mode. The user must make sure that this pin is tied at low level to ensure normal operating conditions. Other values associated with this pin are reserved for manufacturing test.

### 15.6.2 EmbeddedICE

The Cortex-A5 EmbeddedICE-RT is supported via the ICE/JTAG port. It is connected to a host computer via an ICE interface. The internal state of the Cortex-A5 is examined through an ICE/JTAG port which allows instructions to be serially inserted into the pipeline of the core without using the external data bus. Therefore, when in debug state, a store-multiple (STM) can be inserted into the instruction pipeline. This exports the contents of the Cortex-A5 registers. This data can be serially shifted out without affecting the rest of the system.

There are two scan chains inside the Cortex-A5 processor which support testing, debugging, and programming of the EmbeddedICE-RT. The scan chains are controlled by the ICE/JTAG port.

EmbeddedICE mode is selected when JTAGSEL is low. It is not possible to switch directly between ICE and JTAG operations. A chip reset must be performed after JTAGSEL is changed.

For further details on the EmbeddedICE-RT, see the Arm document *ARM IHI 0031A_ARM_debug_interface_v5.pdf*

### 15.6.3 JTAG Signal Description

TMS is the Test Mode Select input which controls the transitions of the test interface state machine.

TDI is the Test Data Input line which supplies the data to the JTAG registers (Boundary Scan Register, Instruction Register, or other data registers).

TDO is the Test Data Output line which is used to serially output the data from the JTAG registers to the equipment controlling the test. It carries the sampled values from the boundary scan chain (or other JTAG registers) and propagates them to the next chip in the serial test circuit.

NTRST (optional in IEEE Standard 1149.1) is a Test-ReSeT input which is mandatory in ARM cores and used to reset the debug logic. On Cortex-A5-based cores, NTRST is a Power On Reset output. It is asserted on power on. If necessary, the user can also reset the debug logic with the NTRST pin assertion during 2.5 MCK periods.

TCK is the Test ClocK input which enables the test interface. TCK is pulsed by the equipment controlling the test and not by the tested device. It can be pulsed at any frequency.

### 15.6.4 IEEE 1149.1 JTAG Boundary Scan

IEEE 1149.1 JTAG Boundary Scan allows pin-level access independent of the device packaging technology.

IEEE 1149.1 JTAG Boundary Scan is enabled when JTAGSEL is high. The SAMPLE, EXTEST and BYPASS functions are implemented. In ICE debug mode, the Arm processor responds with a non-JTAG chip ID that identifies the processor to the ICE system. This is not IEEE 1149.1 JTAG-compliant.

It is not possible to switch directly between JTAG and ICE operations. A chip reset must be performed after JTAGSEL is changed.

A Boundary-scan Descriptor Language (BSDL) file is provided for test setup.

## 15.7 Boundary JTAG ID Register

**Name:** Boundary JTAG ID Register
**Property:** Read-only

JTAG ID Code value is 0x05B3F03F.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | VERSION[3:0] | | | | PART NUMBER[15:12] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PART NUMBER[11:4] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PART NUMBER[3:0] | | | | MANUFACTURER IDENTITY[10:7] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MANUFACTURER IDENTITY[6:0] | | | | | | | 1 |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

**Bits 31:28 – VERSION[3:0]** Product Version Number
Set to 0x0.

**Bits 27:12 – PART NUMBER[15:0]** Product Part Number
Product part number is 0x5B3F.

**Bits 11:1 – MANUFACTURER IDENTITY[10:0]**
Set to 0x01F.

**Bit 0 – 1**
Required by IEEE Std. 1149.1. Set to 1.

## 15.8 Cortex-A5 DP Identification Code Register IDCODE

The Identification Code Register is always present on all DP implementations. It provides identification information about the Arm Debug Interface.

### 15.8.1 JTAG Debug Port (JTAG-DP)

**Name:** JTAG Debug Port (JTAG-DP)
**Property:** Read-only

Debug Port JTAG IDCODE value is 0x5BA00477.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | VERSION[3:0] | | | | PART NUMBER[15:12] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PART NUMBER[11:4] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PART NUMBER[3:0] | | | | DESIGNER[10:7] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DESIGNER[6:0] | | | | | | | 1 |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

**Bits 31:28 – VERSION[3:0]** Product Version Number
Set to 0x5.

**Bits 27:12 – PART NUMBER[15:0]** Product Part Number
Product part number is 0xBA00.

**Bits 11:1 – DESIGNER[10:0]**
Set to 0x23B.

**Bit 0 – 1**
Required by IEEE Std. 1149.1. Set to 1.

### 15.8.2 Serial Wire Debug Port (SW-DP)

**Name:** Serial Wire Debug Port (SW-DP)
**Property:** Read-only

Debug Port Serial Wire IDCODE is 0x5ba02477.

At address 0x0 on read operations when the APnDP bit = 0. Access to the Identification Code Register is not affected by the value of the CTRLSEL bit in the Select Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | VERSION[3:0] | | | | PART NUMBER[15:12] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PART NUMBER[11:4] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PART NUMBER[3:0] | | | | DESIGNER[10:7] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DESIGNER[6:0] | | | | | | | 1 |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

**Bits 31:28 – VERSION[3:0]** Product Version Number
Set to 0x0.

**Bits 27:12 – PART NUMBER[15:0]** Product Part Number
Product part number is 0xBA01.

**Bits 11:1 – DESIGNER[10:0]**
Set to 0x23B.

**Bit 0 – 1**
Required by IEEE Std. 1149.1. Set to 1.

# 16. Standard Boot Strategies

## 16.1 Description

The system always boots from the ROM memory at address 0x0.

The ROM code is a boot program contained in the embedded ROM. It is also called “First level boot loader”.

This microcontroller can be configured to run a Standard Boot mode or a Secure Boot mode. More information on how the Secure Boot mode can be enabled, and how the chip operates in this mode, is provided in the document *SAMA5D2x Secure Boot Strategy*, document no. 44040. To obtain this application note and additional information about the secure boot and related tools, contact a Microchip sales representative.

By default, the chip starts in Standard Boot mode.

## 16.2 Chip Access Using JTAG Connection

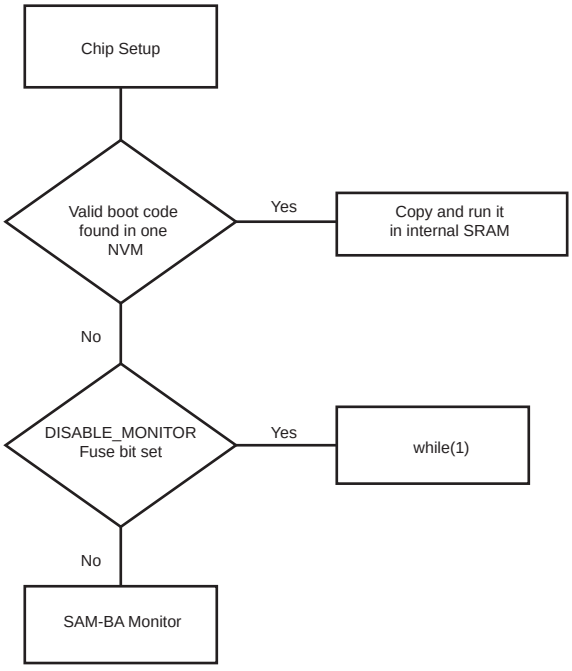
The JTAG connection is disabled at reset, and during the ROM Code execution.

It is re-enabled when the ROM code jumps in the boot file copied from an external Flash memory into the internal SRAM, or when the ROM code launches the SAM-BA monitor, when no boot file has been found in any external Flash memory.

## 16.3 Flow Diagram

The ROM code global flow is shown in the figure below.

**Figure 16-1. ROM Code Flow Diagram**



## 16.4 Chip Setup

When the chip is powered on, the processor clock (PCK) and the master clock (MCK) source is the 12 MHz fast RC oscillator.

The ROM code performs a low-level initialization that follows the steps described below:

1. Stack Setup for Arm supervisor mode.
2. PLLA Initialization
3. Master Clock Selection: when the PLLA is stabilized, the Master Clock source is switched from internal 12 MHz RC to PLLA. The PMC Status Register is polled to wait for MCK Ready. PCK and MCK are now the Main Clock.
4. C Variable Initialization: non zero-initialized data is initialized in the RAM (copy from ROM to RAM). Zero-initialized data is set to 0 in the RAM.

For clock frequencies, see the table Clock Frequencies during External Memory Boot Sequence.

**Note:** No external crystal or clock is needed during the external boot memories sequence. An external clock source is checked before the launch of the SAM-BA monitor to get a more accurate clock signal for USB.

## 16.5 Boot Configuration

The boot sequence is controlled using a Boot Configuration Word in the Fuse area or in the backup registers BUREG.

### 16.5.1 Boot Configuration Word

The Boot Configuration Word allows several customizations of the Boot Sequence:

- To configure the IO Set where the external memories used to boot are connected (see Hardware and Software Constraints for a description of the IO sets)
- To disable the boot on selected memories
- To configure the UART port used as a terminal console
- To configure the JTAG pins used for debug

See the section Boot Configuration Word for a detailed description of all the bitfields in this word.

By default, the value of this word is 0x0.

For MRL A and B parts, the ROM code does not try to detect a valid bootable software in any external memory, and runs directly the SAM-BA monitor. See the figure NVM Bootloader Program Description for MRL A and MRL B Parts.

For MRL C parts, the ROM code only tries to boot on SDMMC1 and SDMMC0 memory interfaces and then run the SAM-BA monitor. See the figure NVM Bootloader Program Description for MRL C Parts.

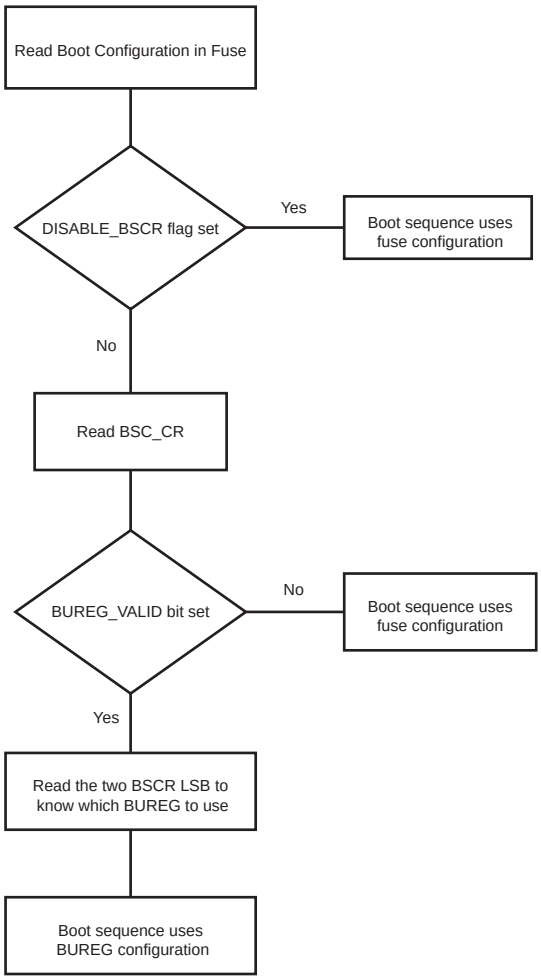
During prototyping phases, the value of this fuse word can be overridden by the content of a backup register. The conditions to enable this feature are as follows:

- The fuse bit DISABLE_BSCR must not be set (default value).
- The Boot Sequence Controller Configuration register (BSC_CR) must have the BUREG_VALID bit set and indicate in BUREG_INDEX which register has to be used.

Using BUREG allows the user to test several boot configuration options, including Secure Boot Mode, without burning fuses.

**Note:** VDDBU must be connected in order to benefit from this feature. However, in production, it is highly recommended to disable this feature and to write the boot configuration in fuses.

**Figure 16-2. Boot Configuration Loading**



### 16.5.2 Boot Sequence Controller Configuration Register

**Name:** BSC_CR
**Offset:** 0xF8048054
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | BUREG_VALID | BUREG_INDEX[1:0] | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | | | |

**Bits 31:16 – WPKEY[15:0]** Write Protect Key (Write-only)

| Value | Name | Description |
|---|---|---|
| 0x6683 | PASSWD | Writing any other value in this field aborts the write operation of the BOOT field. Always reads as 0. |

**Bit 2 – BUREG_VALID** Validate the data in BUREG_INDEX field

| Value | Description |
|---|---|
| 0 | No BUREG contains valid Boot Configuration data. |
| 1 | The BUREG indicated in BUREG_INDEX contains Boot Configuration data for use in configuring the boot sequence. |

**Bits 1:0 – BUREG_INDEX[1:0]** Select the BUREG where the Boot Configuration data must be read

| Value | Name | Description |
|---|---|---|
| 0 | BUREG_0 | Use BUREG 0 value |
| 1 | BUREG_1 | Use BUREG 1 value |
| 2 | BUREG_2 | Use BUREG 2 value |
| 3 | BUREG_3 | Use BUREG 3 value |

### 16.5.3 Backup Registers (BUREG)

The four BUREGs used to override the Boot Configuration Word in Fuse are at addresses:

- 0xF8045400
- 0xF8045404
- 0xF8045408
- 0xF804540C

### 16.5.4 Boot Configuration Word

**Name:** Boot Configuration Word
**Reset:** 0x00000000

The Boot Configuration Word comprises the 32 boot configuration bits (see the table Customer Fuse Matrix).

**WARNING** To avoid any malfunctioning, the user must not write the “DO NOT USE (DNU)” fuse bits.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | SECURE_ MODE | DNU | DNU | DNU | DNU | DISABLE_MONITOR |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DNU | DISABLE_BSCR | QSPI_XIP_MODE | DNU | DNU | EXT_MEM_BOOT_ENABLE | JTAG_IO_SET[1:0] | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UART_CONSOLE[3:0] | | | | SDMMC_1 | SDMMC_0 | NFC[1:0] | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SPI_1[1:0] | | SPI_0[1:0] | | QSPI_1[1:0] | | QSPI_0[1:0] | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 29 – SECURE_ MODE** Enable Secure Boot Mode

| Value | Description |
|---|---|
| 0 | Standard boot sequence |
| 1 | Secure boot sequence |

**Bits 25, 26, 27, 28 – DNU** DO NOT USE

**Bit 24 – DISABLE_MONITOR** Disable SAM-BA Monitor

| Value | Description |
|---|---|
| 0 | If no boot file found, launch SAM-BA Monitor. |
| 1 | SAM-BA Monitor never launched. |

**Bit 23 – DNU** DO NOT USE

**Bit 22 – DISABLE_BSCR** Disable Read of BSC_CR

| Value | Description |
|---|---|
| 0 | If the BUREG index in the BSC_CR is valid, its data replace Fuse configuration bits. |
| 1 | Does not read BSC_CR content, so the Boot settings are those from the Fuse. |

**Bit 21 – QSPI_XIP_MODE** Enable XIP Mode on QSPI Flash

| Value | Description |
|---|---|
| 0 | QSPI is accessed in QSPI mode and data copied into internal SRAM. |
| 1 | QSPI is accessed in XIP mode, and the bootstrap directly executed from it. |

**Bits 19, 20 – DNU** DO NOT USE

**Bit 18 – EXT_MEM_BOOT_ENABLE** Enable Boot on External Memories

| Value | Description |
|---|---|
| 0 | No external memory boot performed. |
| 1 | External memory boot enabled. |

**Bits 17:16 – JTAG_IO_SET[1:0]** Pin Selection for JTAG Access
Refer to "JTAG_TCK on IOSET 4 pin has a wrong configuration after boot" in the document *SAMA5D2 Family Silicon Errata and Data Sheet Clarification*, available on www.microchip.com.

| Value | Name | Description |
|---|---|---|
| 0 | JTAG_IOSET_1 | Use JTAG IO Set 1 |
| 1 | JTAG_IOSET_2 | Use JTAG IO Set 2 |
| 2 | JTAG_IOSET_3 | Use JTAG IO Set 3 |
| 3 | JTAG_IOSET_4 | Use JTAG IO Set 4 |

**Bits 15:12 – UART_CONSOLE[3:0]** Selects the pins and UART interface used as a console terminal

| Value | Name | Description |
|---|---|---|
| 0 | UART_1_IOSET_1 | Use UART1 IO Set 1 |
| 1 | UART_0_IOSET_1 | Use UART0 IO Set 1 |
| 2 | UART_1_IOSET_2 | Use UART1 IO Set 2 |
| 3 | UART_2_IOSET_1 | Use UART2 IO Set 1 |
| 4 | UART_2_IOSET_2 | Use UART2 IO Set 2 |
| 5 | UART_2_IOSET_3 | Use UART2 IO Set 3 |
| 6 | UART_3_IOSET_1 | Use UART3 IO Set 1 |
| 7 | UART_3_IOSET_2 | Use UART3 IO Set 2 |
| 8 | UART_3_IOSET_3 | Use UART3 IO Set 3 |
| 9 | UART_4_IOSET_1 | Use UART4 IO Set 1 |
| 10 | DISABLED | No console terminal |
| 11 | DISABLED | No console terminal |
| 12 | DISABLED | No console terminal |
| 13 | DISABLED | No console terminal |
| 14 | DISABLED | No console terminal |
| 15 | DISABLED | No console terminal |

**Bit 11 – SDMMC_1** Disable SDCard/e.MMC Boot on SDMMC_1
After the first boot, the boot on SDMMC_1 can be disabled by setting this bit.

| Value | Description |
|---|---|
| 0 | Boots on SDMMC_1 using SDMMC_1 PIO Set 1. |
| 1 | Disables boot on SDMMC_1. |

**Bit 10 – SDMMC_0** Disable SDCard/e.MMC Boot on SDMMC_0
After the first boot, the boot on SDMMC_0 can be disabled by setting this bit.

| Value | Description |
|---|---|
| 0 | Boots on SDMMC_0 using SDMMC_0 PIO Set 1. |
| 1 | Disables boot on SDMMC_0. |

**Bits 9:8 – NFC[1:0]** Select the PIO Set Used for NFC Boot

| Value | Name | Description |
|---|---|---|
| 0 | NFC_IOSET_1 | Use NFC IO Set 1 |
| 1 | NFC_IOSET_2 | Use NFC IO Set 2 |
| 2 | DISABLED | NFC boot is disabled |
| 3 | DISABLED | NFC boot is disabled |

**Bits 7:6 – SPI_1[1:0]** Select the PIO Set Used for SPI_1 Boot

| Value | Name | Description |
|---|---|---|
| 0 | SPI_1_IOSET_1 | Use SPI_1 IO Set 1 |
| 1 | SPI_1_IOSET_2 | Use SPI_1 IO Set 2 |
| 2 | SPI_1_IOSET_3 | Use SPI_1 IO Set 3 |
| 3 | DISABLED | SPI boot is disabled |

**Bits 5:4 – SPI_0[1:0]** Select the PIO Set Used for SPI_0 Boot

| Value | Name | Description |
|---|---|---|
| 0 | SPI_0_IOSET_1 | Use SPI_0 IO Set 1 |
| 1 | SPI_0_IOSET_2 | Use SPI_0 IO Set 2 |
| 2 | DISABLED | SPI boot is disabled |
| 3 | DISABLED | SPI boot is disabled |

**Bits 3:2 – QSPI_1[1:0]** Select the PIO Set Used for QSPI_1 Boot

| Value | Name | Description |
|---|---|---|
| 0 | QSPI_1_IOSET_1 | Use QSPI_1 PIO Set 1 |
| 1 | QSPI_1_IOSET_2 | Use QSPI_1 PIO Set 2 |
| 2 | QSPI_1_IOSET_3 | Use QSPI_1 PIO Set 3 |
| 3 | DISABLED | QSPI_1 boot is disabled |

**Bits 1:0 – QSPI_0[1:0]** Select the PIO Set Used for QSPI_0 Boot

| Value | Name | Description |
|---|---|---|
| 0 | QSPI_0_IOSET_1 | Use QSPI_0 PIO Set 1 |
| 1 | QSPI_0_IOSET_2 | Use QSPI_0 PIO Set 2 |
| 2 | QSPI_0_IOSET_3 | Use QSPI_0 PIO Set 3 |

© 2018 Microchip Technology Inc.

| Value | Name | Description |
|---|---|---|
| 3 | DISABLED | QSPI_0 boot is disabled |

### 16.5.5 NVM Boot Sequence

The ROM code performs the initialization and valid code detection for the external memories as described below only if those memories are not disabled in the Boot Configuration word.

**Figure 16-3. NVM Bootloader Program Description for MRL A and MRL B Parts**

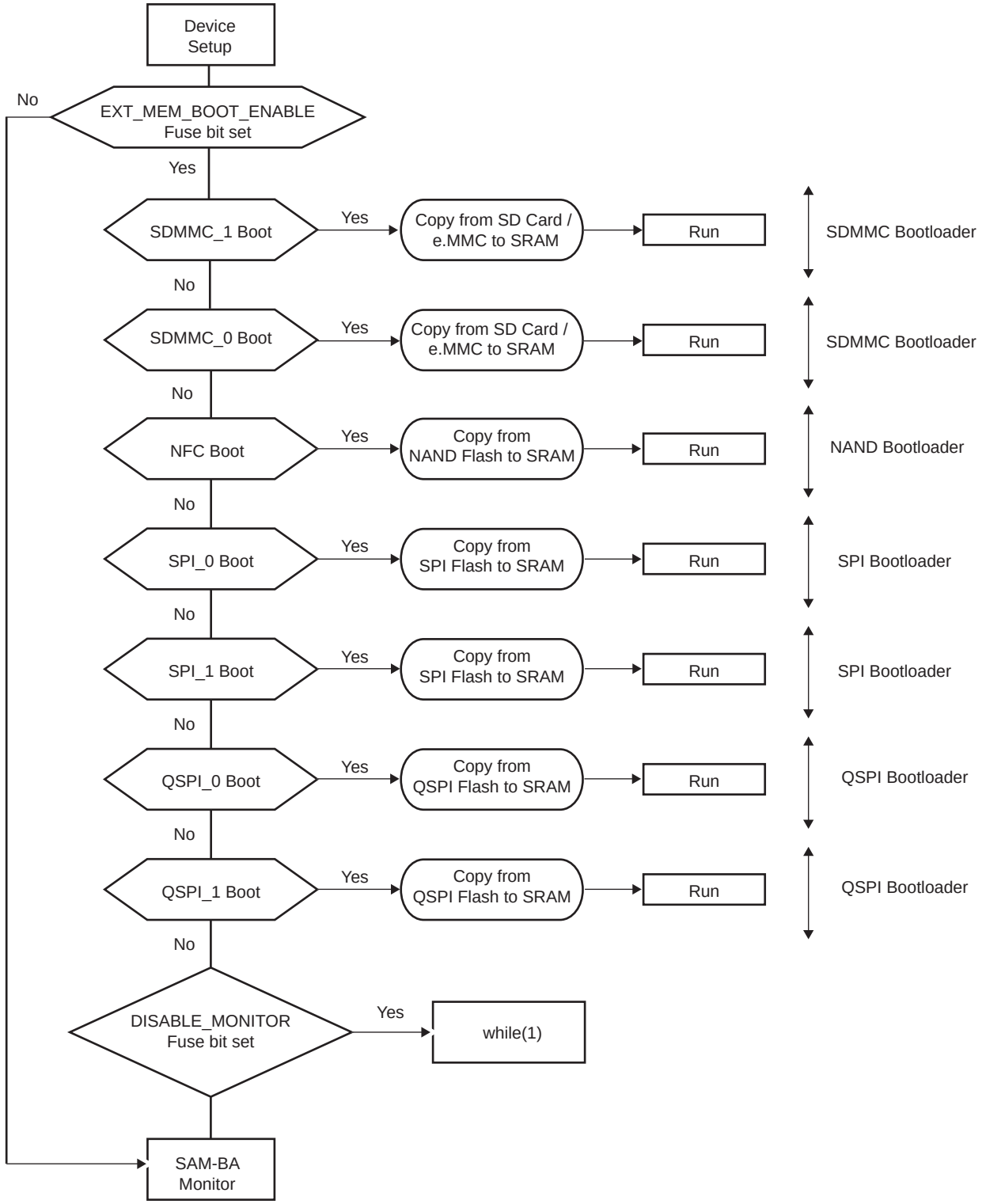


**Figure 16-4. NVM Bootloader Program Description for MRL C Parts**

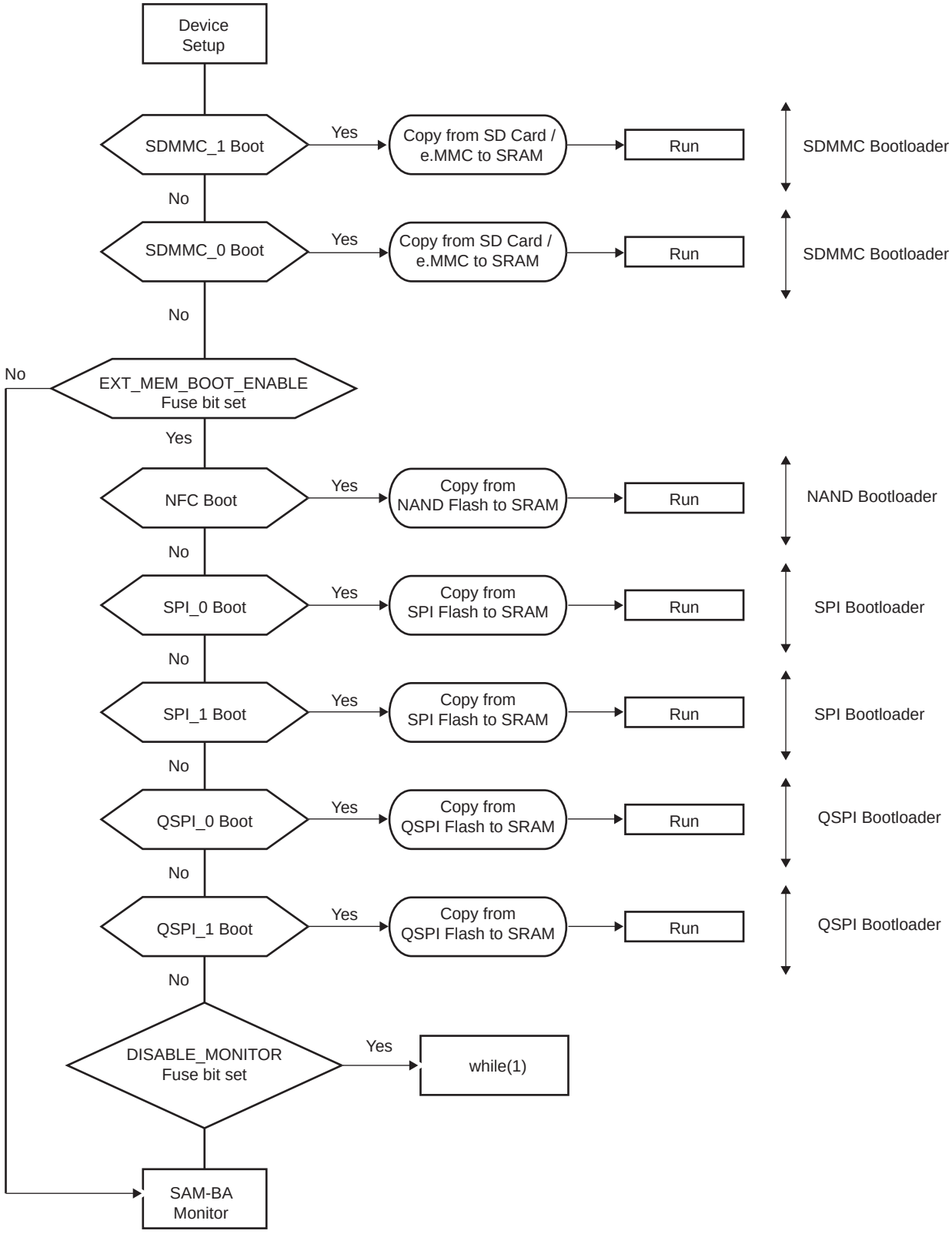
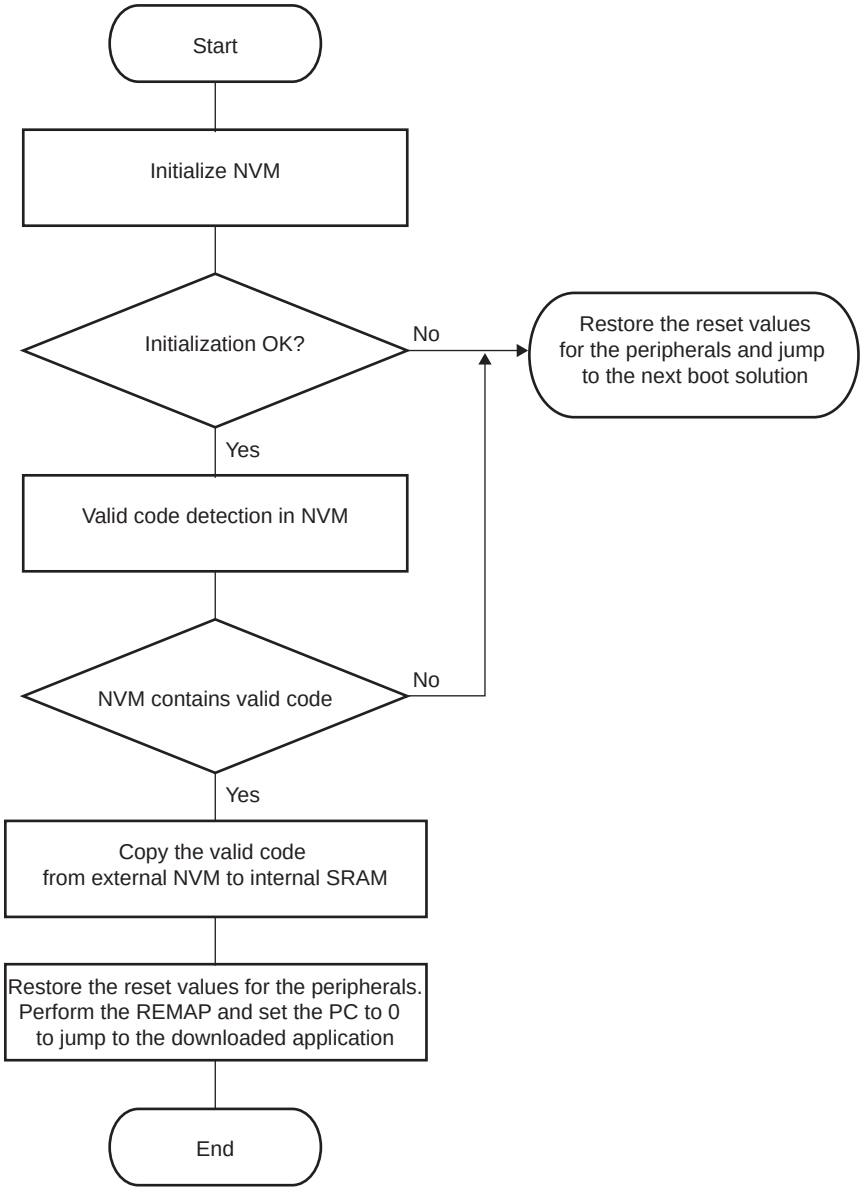


**Figure 16-5. NVM Boot Diagram**



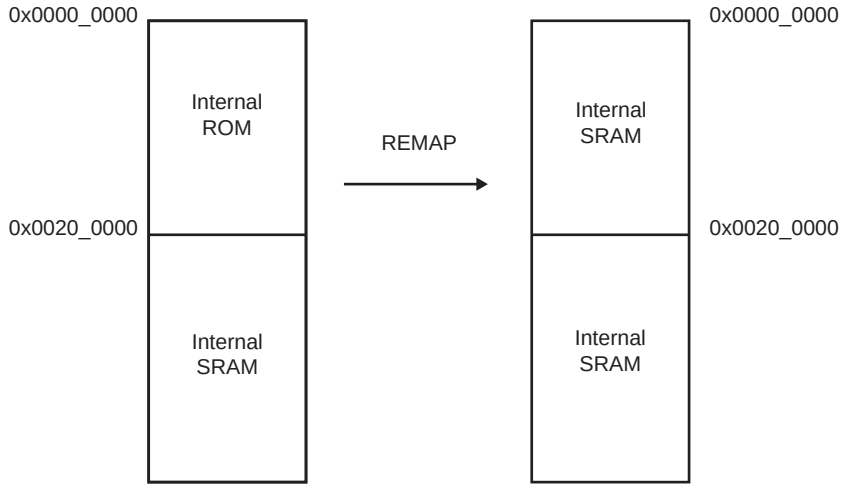

The NVM bootloader program first initializes the PIOs related to the NVM device. Then it configures the right peripheral depending on the NVM and tries to access this memory. If the initialization fails, it restores the reset values for the PIO and the peripheral, and then tries to fulfill the same operations on the next NVM of the sequence.

If the initialization is successful, the NVM bootloader program reads the beginning of the NVM and determines if the NVM contains a valid code.

If the NVM does not contain a valid code, the NVM bootloader program restores the reset value for the peripherals and then tries to fulfill the same operations on the next NVM of the sequence.

If a valid code is found, this code is loaded from the NVM into the internal SRAM and executed by branching at address 0x0000_0000 after remap. This code may be the application code or a second-level bootloader. All the calls to functions are PC-relative and do not use absolute addresses.

**Figure 16-6. Remap Action after Download Completion**




### 16.5.6 Valid Code Detection

There are two kinds of valid code detection, which are described in the following sections.

#### 16.5.6.1 Arm Exception Vectors Check

The NVM bootloader program reads and analyzes the first 28 bytes corresponding to the first seven Arm exception vectors. Except for the sixth vector, these bytes must implement the Arm instructions for either branch or load PC with PC-relative addressing.

**Figure 16-7. LDR Opcode**

| 31 28 | 27 24 | 23 20 | 19 16 | 15 12 | 11 0 |
|---|---|---|---|---|---|
| 1 1 1 0 | 0 1 I P | U 1 W 0 | Rn | Rd | Offset |

**Figure 16-8. B Opcode**

| 31 28 | 27 24 | 23 0 |
|---|---|---|
| 1 1 1 0 | 1 0 1 0 | Offset (24 bits) |

Unconditional instruction: 0xE for bits 31 to 28. Load PC with the PC-relative addressing instruction:

- Rn = Rd = PC = 0xF
- I==0 (12-bit immediate value)
- P==1 (pre-indexed)
- U offset added (U==1) or subtracted (U==0)
- W==1

The sixth vector, at the offset 0x14, contains the size of the image to download. The user must replace this vector with the user's own vector. This procedure is described below.

**Figure 16-9. Arm Vector 6 Structure**

| 31 0 |
|---|
| Size of the code to download in bytes |

The value has to be smaller than 64 Kbytes.

An example of valid vectors:

```
00          ea000006                B 0x20
04          eafffffe                B 0x04
```

```
08          ea00002f                B _main
0c          eafffffe                B 0x0c
10          eafffffe                B 0x10
14          00001234                B 0x14 ← Code size = 4660 bytes
18          eafffffe                B 0x18
```

#### 16.5.6.2 boot.bin File Check

This method is the one used on FAT-formatted SD Card and e.MMC. The boot program must be a file named `boot.bin` written in the root directory of the file system. Its size must not exceed the maximum size allowed of 64 Kbytes (0x10000).

### 16.5.7 Detailed Memory Boot Procedures

#### 16.5.7.1 NAND Flash Boot: NAND Flash Detection

After the NAND Flash interface configuration, a reset command is sent to the memory.

Hardware ECC detection and correction are provided by the PMECC peripheral. See the section PMECC Controller Functional Description for more details.

The Boot Program is able to retrieve NAND Flash parameters and ECC requirements using two methods as follows:

- The detection of a specific header written at the beginning of the first page of the NAND Flash

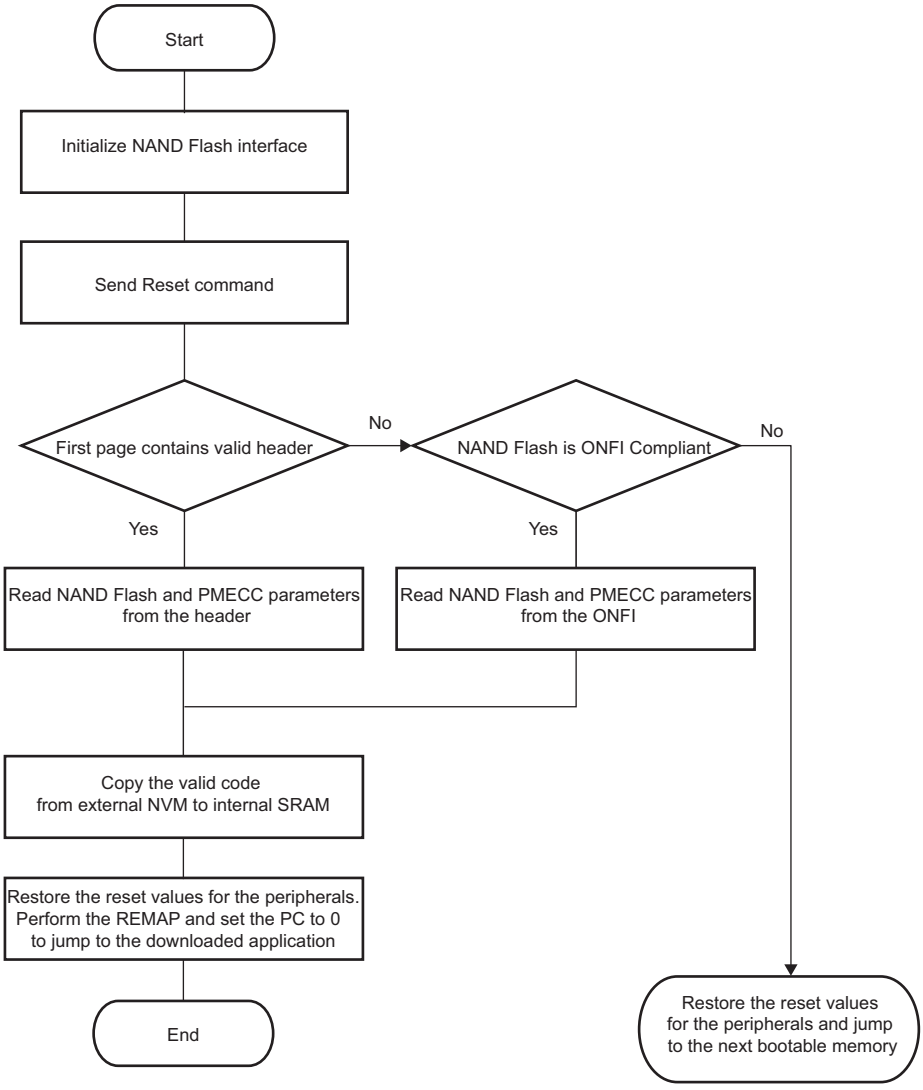
or

- Through the ONFI parameters for the ONFI-compliant memories

However, it is highly recommended to use the NAND Flash Header method (first bullet above) since it indicates exactly how the PMECC has been configured to write the bootable program in the NAND Flash, and not to rely only on the NAND Flash capabilities.

**Note:** Booting on 16-bit NAND Flash is not possible; only 8-bit NAND Flash memories are supported.

**Figure 16-10. Boot NAND Flash Download**




#### 16.5.7.1.1 NAND Flash Specific Header Detection (Recommended Solution)

This is the first method used to determine NAND Flash parameters. After Initialization and Reset command, the Boot Program reads the first page without an ECC check, to determine whether the NAND parameter header is present. The header is made of 52 times the same 32-bit word (for redundancy reasons) which must contain NAND and PMECC parameters used to correctly perform the read of the rest of the data in the NAND. This 32-bit word is described below.

If the header is valid, the Boot Program continues with the detection of a valid code.

**Note:** Booting on 16-bit NAND Flash is not possible; only 8-bit NAND Flash memories are supported.

***[NAND Flash PMECC Register]***

**Name:** NAND Flash PMECC Register

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | key[3:0] | | | | | eccOffset[8:6] | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | eccOffset[5:0] | | | | | | sectorSize[1:0] | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | eccBitReq[2:0] | | | spareSize[8:4] | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | spareSize[3:0] | | | | nbSectorPerPage[2:0] | | | usePMECC |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 31:28 – key[3:0]** Value 0xC Must be Written here to Validate the Content of the Whole Word.

**Bits 26:18 – eccOffset[8:0]** Offset of the First ECC Byte in the Spare Zone
A value below 2 is not allowed and is considered as 2.

**Bits 17:16 – sectorSize[1:0]** Size of the ECC Sector

| Value | Description |
|---|---|
| 0 | For 512 bytes |
| 1 | For 1024 bytes per sector |
| Other values | For future use |

**Bits 15:13 – eccBitReq[2:0]** Number of ECC Bits Required

| Value | Description |
|---|---|
| 0 | 2-bit ECC |
| 1 | 4-bit ECC |
| 2 | 8-bit ECC |
| 3 | 12-bit ECC |
| 4 | 24-bit ECC |
| 5 | 32-bit ECC |

**Bits 12:4 – spareSize[8:0]** Size of the Spare Zone in Bytes

**Bits 3:1 – nbSectorPerPage[2:0]** Number of Sectors per Page

**Bit 0 – usePMECC** Use PMECC

| Value | Description |
|---|---|
| 0 | Do not use PMECC to detect and correct the data. |
| 1 | Use PMECC to detect and correct the data. |

***ONFI 2.2 Parameters (Not Recommended)***

In case no valid header is found, the Boot Program checks if the NAND Flash is ONFI-compliant, sending a Read Id command (0x90) with 0x20 as parameter for the address. If the NAND Flash is ONFI-compliant, the Boot Program retrieves the following parameters with the help of the Get Parameter Page command:

- Number of bytes per page (byte 80)
- Number of bytes in spare zone (byte 84)
- Number of ECC bit corrections required (byte 112)
- ECC sector size: by default, set to 512 bytes; or to 1024 bytes if the ECC bit capability above is 0xFF

By default, the ONFI NAND Flash detection turns ON the usePmecc parameter, and the ECC correction algorithm is automatically activated.

Once the Boot Program retrieves the parameter, using one of the two methods described above, it reads the first page again, with or without ECC, depending on the usePmecc parameter. Then it looks for a valid code programmed just after the header offset 0xD0. If the code is valid, the program is copied at the beginning of the internal SRAM.

**Note:** Booting on 16-bit NAND Flash is not possible; only 8-bit NAND Flash memories are supported.

#### 16.5.7.2 NAND Flash Boot: PMECC Error Detection and Correction

NAND Flash boot procedure uses PMECC to detect and correct errors during NAND Flash read operations in two cases:

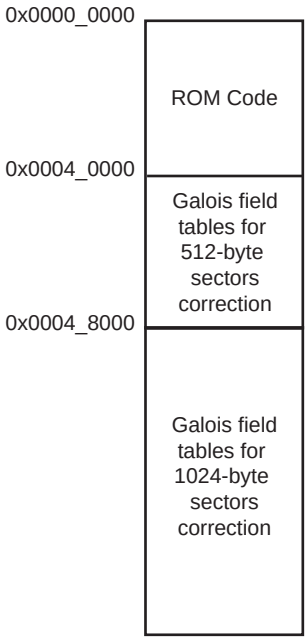
- When the usePmecc flag is set in a specific NAND header.
- If the flag is not set, no ECC correction is performed during the NAND Flash page read. When the NAND Flash has been detected using ONFI parameters.

The ROM memory embeds the Galois field tables. The user does not need to embed them in his own software.

The Galois field tables are mapped in the ROM just after the ROM code, as illustrated in the figure below.

For a full description and an example of how to use the PMECC detection and correction feature, see the software package dedicated to this device on our website.

**Figure 16-11. Galois Field Table Mapping**




#### 16.5.7.3 SDCard/e.MMC Boot

The SDCard/e.MMC boot requires the Card Detect pin to be connected. If the level on the Card Detect pin is low, SDCard/e.MMC access is initiated (IOs toggling). If not, no communication with SDCard/e.MMC is performed (no IOs toggling).

The SDMMC0 and SDMMC1 pin card detect must be left unconnected if the interfaces are used with a non-removable and non-bootable device (wifi module, etc.).

This prevents the ROM code from trying to boot out of these interfaces and avoids a bad behavior of the boot.

In the case of non-removable devices (soldered on board), the card detect can be managed by software (refer to "Force Card Detect" bit) or by hardware by enabling the pull-down resistor on SDMMCx_CD PIO after the ROM Code execution..

**Supported SDCard Devices**

SDCard Boot supports all SDCard memories compliant with the SD Memory Card Specification V3.0. This includes SDMMC cards.

**e.MMC with Boot Partition**

The ROM code first checks if the e.MMC Boot Partition is enabled. If enabled, the ROM code reads the first 64 Kbytes of the boot partition, and copy them into the internal SRAM.

**FAT Filesystem boot**

If no boot partition is enabled on an e.MMC, the boot process continues with a Standard SDCard/e.MMC detection, and the ROM code looks for a `boot.bin` file in the root directory of a FAT12/16/32 file system.

#### 16.5.7.4 SPI Flash Boot

Two types of SPI Flash are supported

- SPI DataFlash

- SPI Serial Flash

The SPI Flash bootloader tries to boot on SPI0, first looking for SPI Serial Flash, and then for SPI DataFlash.

It uses only one valid code detection: analysis of Arm exception vectors.

The SPI Flash read is done by means of a Continuous Read command from the address 0x0. This command is 0xE8 for DataFlash and 0x0B for Serial Flash devices.

#### 16.5.7.4.1 Supported DataFlash Devices

The SPI Flash Boot program supports the DataFlash devices listed in the table below.

| Device | Density | Page Size (bytes) | Number of Pages |
|---|---|---|---|
| AT45DB011 | 1 Mbit | 264 | 512 |
| AT45DB021 | 2 Mbits | 264 | 1024 |
| AT45DB041 | 4 Mbits | 264 | 2048 |
| AT45DB081 | 8 Mbits | 264 | 4096 |
| AT45DB161 | 16 Mbits | 528 | 4096 |
| AT45DB321 | 32 Mbits | 528 | 8192 |
| AT45DB642 | 64 Mbits | 1056 | 8192 |
| AT45DB641 | 64 Mbits | 264 | 37768 |

#### 16.5.7.4.2 Supported Serial Flash Devices

The SPI Flash Boot program supports all SPI Serial Flash devices responding correctly to both Get Status and Continuous Read commands.

### 16.5.7.5 QSPI NOR Flash Boot for MRL A and MRL B

**Important:** This section applies to the devices listed in the table below.

**Table 16-1. SAMA5D2 MRL A and MRL B Parts**

| Device Name |
|---|
| ATSAMA5D22A |
| ATSAMA5D24A |
| ATSAMA5D27A |
| ATSAMA5D28A |
| ATSAMA5D21B |
| ATSAMA5D22B |
| ATSAMA5D23B |
| ATSAMA5D24B |

| .....continued |
|---|
| **Device Name** |
| ATSAMA5D26B |
| ATSAMA5D27B |
| ATSAMA5D28B |

#### 16.5.7.5.1 Definitions (MRL A, MRL B)

SPI x-y-z protocol:

- Command opcode is sent on x I/O data line(s) with x in {1, 2, 4}
- Address is sent on y I/O data line(s) with y in {1, 2, 4}
- Data are sent or received on z I/O data lin(s) with z in {1, 2, 4}

Relevant combinations are shown in the table below:

| Protocol | Description |
|---|---|
| SPI 1-1-1 | Legacy SPI protocol using MOSI/IO0 and MISO/IO1 lines |
| SPI 1-1-2 | SPI Dual Output using IO0 and IO1 lines |
| SPI 1-2-2 | SPI Dual I/O using IO0 and IO1 lines |
| SPI 2-2-2 | SPI Dual Command using IO0 and IO1 lines |
| SPI 1-1-4 | SPI Quad Output using IO0, IO1, IO2 and IO3 lines |
| SPI 1-4-4 | SPI Quad I/O using IO0, IO1, IO2 and IO3 lines |
| SPI 4-4-4 | Quad Command using IO0, IO1, IO2 and IO3 lines |

#### 16.5.7.5.2 Supported QSPI Memory Manufacturers (MRL A, MRL B)

The ROM code only supports the three following manufacturers (manufacturer ID):

- Cypress (01h)
- Micron (20h)
- Macronix (C2h)

Other manufacturer IDs are ignored and the ROM code jumps to the next non-volatile memory in the Boot Sequence.

#### 16.5.7.5.3 SPI Clock Frequency, Phase and Polarity (MRL A, MRL B)

The peripheral clock of each QSPI controller is gated from the Master Clock (MCK). The ROM code configures MCK and the QSPI Serial Clock (QSCK). See the table "Clock Frequencies during External Memory Boot Sequence".

The QSPI controller is configured to use Clock Mode 0: Both CPHA and CPOL are cleared in QSPI_SCR.

- CPOL = 0: The inactive state value of QSCK is logic level zero.
- CPHA = 0: Data is captured on the leading edge of QSCK and changed on the following edge of QSCK.

#### 16.5.7.5.4 QSPI Memory Detection (MRL A, MRL B)

The ROM code probes the QSPI memory using JEDEC Read ID commands. However the opcode and the SPI protocol to be used to read the JEDEC ID of the QSPI memory depend on its Manufacturer and its current internal state.

**Cypress**

Cypress memories do not support the SPI 4-4-4 protocol. The command opcode is always sent on the single MOSI/IO1 data line. Hence when writing the 9Fh opcode on MOSI during the first 8 cycles, Cypress memories should always reply on MISO with their JEDEC ID during the following cycles.

**Micron**

Micron memories provide three modes of operation:

- Extended SPI: standard SPI protocol upgraded with dual (SPI 1-1-2, SPI 1-2-2) and quad (SPI 1-1-4, SPI 1-4-4) operations
- Dual I/O SPI: all commands use the SPI 2-2-2 protocol
- Quad I/O SPI: all commands use the SPI 4-4-4 protocol

The ROM code supports the Extended and Quad I/O SPI modes but not Dual I/O SPI.

In Extended SPI mode, Micron memories replies to the regular Read JEDEC ID opcode using the protocol SPI 1-1-1: the 9Fh opcode is sent on MOSI using eight clock cycles then the JEDEC ID is read from MISO only.

In Quad I/O SPI mode, Micron memories no longer reply to the regular Read JEDEC ID (9Fh) but answer the new Read JEDEC ID Multiple I/O command instead: The AFh op code is sent on the 4 I/O lines using 2 clock cycles, then only the 3 first bytes (1 byte for the Manufacturer ID followed by 2 bytes for the Device ID) of the JEDEC ID are returned by the memory on the 4 I/O lines.

The AFh opcode is not supported in Extended SPI mode.

**Macronix**

Macronix memories provide two modes of operation:

- SPI: standard SPI protocol upgraded with dual (SPI 1-1-2, SPI 1-2-2) and quad (SPI 1-1-4, SPI 1-4-4) operations
- QPI: all commands use the SPI 4-4-4 protocol

The ROM code supports only the Macronix SPI mode.

In SPI mode, Macronix memories reply to the regular Read JEDEC ID opcode using the protocol SPI 1-1-1: The 9Fh opcode is sent on MOSI using 8 clock cycles then the JEDEC ID is read from MISO only.

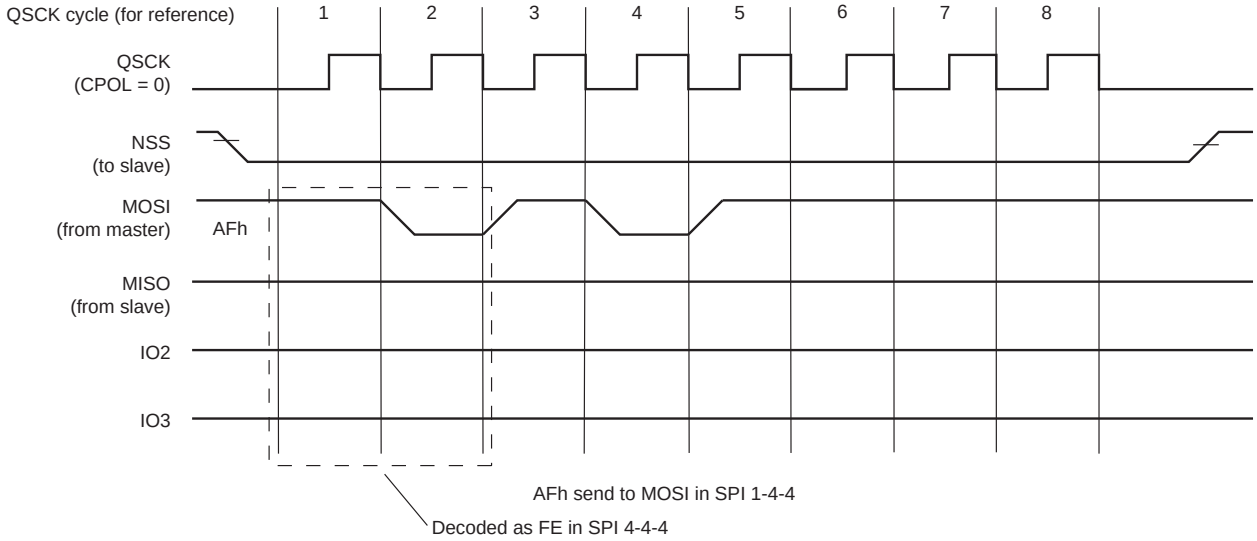
Hence the ROM code uses the following sequence to read the JEDEC ID:

| Step | SPI Protocol | Opcode | Support by Manufacturer Modes |
|---|---|---|---|
| 1 | 1-1-1 | 9Fh | Cypress, Micron Extended SPI, Macronix SPI |
| 2 | 1-4-4 | AFh | **See Note below** |
| 3 | 4-4-4 | AFh | Micron Quad I/O SPI |

**Note:** Step 2 is a wrong combination but should not change the internal state of any QSPI memory. Indeed, **assuming pull-up resistors are used on the four I/O lines**, sending the AFh op code with SPI

1-x-y protocols (the opcode is sent only to MOSI during eight clock cycles) to a memory in Quad I/O SPI or QPI mode should be harmless (FEh opcode decoded by the memory when in Quad I/O SPI or QPI mode: unknown opcode). See the figure below.

**Figure 16-12. QSPI Transfer Format (CPHA = 0, 8 bits per opcode)**




#### 16.5.7.5.5 Allowing Quad I/O Commands (MRL A, MRL B)

On most QSPI memories, some pins are shared between legacy functions such as Write Protect (#WP), Hold (#HOLD) or Reset (#RST) and I/O data lines 2 and 3.

Hence before sending any Quad I/O commands, the ROM code updates the relevant register to reassign those pins to functions IO2 and IO3:

**Cypress**

The ROM code sets the Quad Enable bit (bit1) in the Configuration Register (CR) / Status Register 2 (SR2). The bit is volatile or non-volatile depending on memory versions. This operation is performed using the Write Status command (01h), setting SR1 to 00h and SR2 to 02h.

**Micron**

The ROM code updates the Enhanced Volatile Configuration Register (EVCR) to clear the Quad I/O protocol bit (bit7) hence enabling the Quad I/O protocol. From this point, all commands must use the SPI 4-4-4 protocol.

**Macronix**

The ROM code updates the Status Register (SR1) to set its Quad Enable non-volatile bit (bit6) using the Write Status command (01h).

#### 16.5.7.5.6 Configuration of Fast Read Quad I/O (EBh) Operations (MRL A, MRL B)

The ROM code performs all read operations using the Fast Read Quad I/O (EBh) opcode followed by a 3-byte address.

Since we cannot afford to add an exhaustive table of Read JEDEC IDs and to provide support of future products of memory manufacturers, the ROM code only relies on the very first byte of the JEDEC ID, i.e., the Manufacturer ID, to configure read operations. The ROM code matches the Manufacturer ID as shown in the following table.

**Table 16-2. Fast Read Quad I/O (EBh) Configuration by Manufacturer ID**

| Manufacturer ID | Manufacturer | SPI Protocol | # of Mode Cycles | # of Dummy Cycles | Mode Cycle Value | |
|---|---|---|---|---|---|---|
| | | | | | (no XIP) | (XIP) |
| 01h | Cypress | SPI 1-4-4 | 2 (1) | 4 (1) | 00h | A0h |
| 20h | Micron | SPI 4-4-4 | 1 (2) | 9 (2) | 1h | 0h |
| C2h | Macronix | SPI 1-4-4 | 2 (3) | 4 (3) | 00h | F0h |

**Note:**

1. The ROM code **expects** the Latency Control non-volatile bits of the Cypress Status Register 3 (SR3) / Control Register 1 (CR1) to be zero (LC = 0). The ROM code **does not update** this value.
2. The ROM code sets the number of mode/dummy cycles for Micron memories updating bits [7:4] of their Volatile Configuration Register (VCR) with the 81h opcode. During this update of the VCR:
   - ROM code v1.1 always clears bit3 to enable XIP.
   - ROM code v1.2 clears bit3 to enable XIP if and only if XIP bit is set in the Boot Config word, otherwise it sets bit3 to disable XIP.
3. The ROM code configures the number of mode/dummy cycles for Macronix memories by clearing the volatile DC0 and DC1 bits (bits [7:6]) in the Configuration Register (CR) / Status Register 2 (SR2). It also clears the 4-byte volatile bit (bit5), resulting in the memory going back to its 3-byte address mode. This register updated (read, modify, write) using a Write Status command (01h).

#### 16.5.7.5.7 Miscellaneous Information (MRL A, MRL B)

**Pull-up Resistors**

The ROM code removes the internal pull-up resistors when it configures PIO controller to mux the QSPI controller I/O lines. Therefore the probing step may fail if the Quad I/O mode of the memory has not been enabled yet and if this memory does not embed an internal pull-up resistor on #HOLD or #RESET pin.

This is why we recommend to add external pull-up resistors if needed on the four I/O data lines MOSI/IO0, MISO/IO1, #WP/IO2 and #HOLD/IO3.

Another solution is to update the Quad Enable non-volatile bit in the relevant register to reassign #WP and #HOLD/#RESET pins to functions IO2 and IO3.

**4-byte Address Mode (> 16 MB memories)**

Except for Macronix, the ROM code never sends any command to the memory to leave its 4-byte address mode or to select its first memory bank.

The ROM code expects to read from the very beginning of the QSPI memory using the Fast Read Quad I/O (EBh) command with a 3-byte address. Therefore we recommend that the customer application does not change the internal state of the QSPI memory but uses 4-byte opcodes when needed instead. Hence the ROM code can still read from the QSPI memory after a reset of the SoCs.

#### 16.5.7.6 QSPI NOR Flash Boot for MRL C

**Important:** This section applies to the devices listed in the table below:

| Device Name |
|---|
| ATSAMA5D21C |
| ATSAMA5D22C |
| ATSAMA5D23C |
| ATSAMA5D24C |
| ATSAMA5D26C |
| ATSAMA5D27C |
| ATSAMA5D28C |

##### 16.5.7.6.1 Supported QSPI Memories by Manufacturer (MRL C)

**Table 16-3. QSPI NOR Memories Tested with and Supported by MRL C ROM Code (non exhaustive)**

| Manufacturer | Memories |
|---|---|
| Microchip (SST) | SST26VF016B |
| | SST26VF032B |
| | SST26VF032BA |
| | SST26VF064B |
| Micron | N25Q128A |
| | N25Q128A13ESF |
| | N25Q256A13ESF |
| | N25Q512A13 |
| | MT25QL01G |

| .....continued | |
|---|---|
| **Manufacturer** | **Memories** |
| Macronix | MX25V4035FM2I |
| | MX25V8035FM2I |
| | MX25V1635FM2I |
| | MX25L3233FM2I-08G |
| | MX25L3273FM2I-08G |
| | MX25L6433FM2I-08G |
| | MX25L6473FM2I-08G |
| | MX25L12835FM2I-10G |
| | MX25L12845GMI-08G |
| | MX25L12873GM2I-08G |
| | MX25L25645G |
| | MX25L25673G |
| | MX25L51245GMI-10G |
| | MX66L1G45GMI-08G |
| Spansion | S25FL127 (normal boot only; XIP fails) |
| | S25FL164 |
| | S25FL512 |
| Winbond | Limited support. Refer to Note. |

#### 16.5.7.6.2 Hardware Considerations (MRL C)

The ROM code configures the hardware so that:

- the QSPI controller uses SPI Mode 0 (CPOL = 0 and CPHA = 0),
- the QSPIx_SCK clock frequency is ≤ 50 MHz,
- QSPIx_SCK and QSPIx_CS do not use any internal pull-up/pull-down resistor,
- each QSPIx_IO{0,1,2,3} uses the PIO controller's internal pull-up resistor.

#### 16.5.7.6.3 Software Considerations (MRL C)

Before reading any data, the ROM code sends a software reset to the QSPI NOR memory. Then the ROM code looks for the Serial Flash Discoverable Parameters (SFDP) of the QSPI NOR memory, if available, to learn the parameters (instruction op code, timing settings) required to read the user-programmed boot file.

If SFDP tables are not available, the ROM code uses hard-coded values as fallback settings to read the boot file.

The ROM code supports any QSPI NOR memory which can provide its Serial Flash Discoverable Parameters (SFDP) as defined in the JEDEC JESD216B standard.

The supported revisions of this JEDEC standard are:

- JESD216 (version 1.0)

- JESD216 rev. A (version 1.5)
- JESD216 rev. B (version 1.6)

Refer to the datasheet of the QSPI NOR memory to check compliance with any of the above JEDEC JESD216 standard revisions/versions.

***QSPI NOR memories with SFDP (JEDEC JESD216x compliant)***

The ROM code reads the memory SFDP tables to learn the factory settings (instruction op code, number of dummy cycles, etc.). The ROM code also reads bits[22:20] in DWORD15 from the Basic Flash Parameter Table (refer to JEDEC JESD216B specification) to select and then execute the relevant procedure, if any, to set the Quad Enable (QE) bit in some internal register of the QSPI NOR memory.

For most memory manufacturers, this QE bit is nonvolatile and must be set before performing any Quad SPI command. This is the only persistent setting that the ROM code may change in the internal registers of the QSPI NOR memory. All other settings are kept unchanged.

**Note:** Values 001b and 100b for bits[22:20] in DWORD15 are not correctly supported by ROM code rev. C. Consequently, booting from memories using one the above values in their SFDP tables is likely to fail. Almost all Winbond QSPI NOR memories suffer from this issue.

Refer to the datasheet of the QSPI NOR memory to find which value was chosen by the memory manufacturer and written into the SFDP tables.

Finally, the ROM code reads the boot file from the data area of the QSPI NOR memory, and then continues its boot procedure.

***QSPI NOR memories without SFDP***

This section only applies when the ROM code fails to read the SFDP tables from the QSPI NOR memory.

The ROM code reads the JEDEC ID of the QSPI NOR memory, and then selects the read settings based on the manufacturer ID (first byte of the JEDEC ID) from the following hard-coded values:

| | Cypress (01h) | Micron (20h) | Macronix (C2h) | Winbond (EFh) | Others |
|---|---|---|---|---|---|
| Fast Read protocol | SPI 1-4-4 | SPI 1-4-4 | SPI 1-4-4 | SPI 1-4-4 | SPI 1-1-1 |
| Fast Read op code | EBh | EBh | EBh | EBh | 0Bh |
| Address width | 24 bits | 24 bits | 24 bits | 24 bits | 24 bits |
| Number of mode clock cycles | 2 | 1 | 2 | 2 | 0 |
| Number of wait states | 4 | 9 | 4 | 4 | 8 |
| Value of mode cycles to enter the 0-4-4 mode (XIP) | A0h | 0h<br>The ROM code first sets XIP bit[3] in the Volatile Configuration Register (VCR) | 0Fh | A5h | N/A |
| Value of mode cycles to exit the 0-4-4 mode (normal read) | 00h | 1h | 00h | FFh | N/A |
| XIP supported | Yes | Yes | Yes | Yes | No |

Those hard-coded parameters give a last chance to the ROM code to boot from a QSPI NOR memory in either normal mode or XIP (continuous read) mode.

### 16.5.8 Hardware and Software Constraints

The table below provides clock frequencies configured by the ROM code during boot.

**Table 16-4. Clock Frequencies during External Memory Boot Sequence**

| Clock | MRL A | MRL B | MRL C |
|---|---|---|---|
| PLLA | 792 MHz | 792 MHz | 756 MHz |
| PCK | 396 MHz | 396 MHz | 378 MHz |
| MCK | 132 MHz | 132 MHz | 126 MHz |
| SDMMC (init/operational) | 400 kHz / 25 MHz | 400 kHz / 25 MHz | 400 kHz / 25 MHz |
| SPI | 6 MHz | 12 MHz | 12 MHz |
| QSPI | 25 MHz | 50 MHz | 50 MHz |

The NVM drivers use several PIOs in Peripheral mode to communicate with external memory devices. Care must be taken when these PIOs are used by the application. The connected devices could be unintentionally driven at boot time, and thus electrical conflicts between the output pins used by the NVM drivers and the connected devices could occur.

To ensure the correct functionality, it is recommended to plug in critical devices to other pins not used by the NVM.

The table below contains a list of pins that are driven during the boot program execution. These pins are driven during the boot sequence for a period of less than 1 second if no correct boot program is found. For MRL C parts only, the drive strength of some I/O pins is set to 'medium' while the pins are used in peripheral mode by the ROM code. For MRL A and B, drive strength is always low.

Before performing the jump to the application in the internal SRAM, all the PIOs and peripherals used in the boot program are set to their reset state.

**Table 16-5. PIO Driven during Boot Program Execution**

| NVM Bootloader | Peripheral | IO Set | Pin | PIO Line | Drive Strength (MRL C only) |
|---|---|---|---|---|---|
| SD Card / e.MMC | SDMMC_0 | 1 | SDMMC0_CK | PIOA0 | low |
| | | | SDMMC0_CMD | PIOA1 | medium |
| | | | SDMMC0_DAT0 | PIOA2 | medium |
| | | | SDMMC0_DAT1 | PIOA3 | medium |
| | | | SDMMC0_DAT2 | PIOA4 | medium |
| | | | SDMMC0_DAT3 | PIOA5 | medium |
| | | | SDMMC0_DAT4 | PIOA6 | low |
| | | | SDMMC0_DAT5 | PIOA7 | low |
| | | | SDMMC0_DAT6 | PIOA8 | low |
| | | | SDMMC0_DAT7 | PIOA9 | low |
| | | | SDMMC0_RSTN | PIOA10 | medium |
| | | | SDMMC0_1V8SEL | PIOA11 | low |
| | | | SDMMC0_WP | PIOA12 | medium |
| | | | SDMMC0_CD | PIOA13 | medium |
| | SDMMC_1 | 1 | SDMMC1_DAT0 | PIOA18 | medium |
| | | | SDMMC1_DAT1 | PIOA19 | medium |
| | | | SDMMC1_DAT2 | PIOA20 | medium |
| | | | SDMMC1_DAT3 | PIOA21 | medium |
| | | | SDMMC1_CK | PIOA22 | low |
| | | | SDMMC1_RSTN | PIOA27 | medium |
| | | | SDMMC1_CMD | PIOA28 | medium |
| | | | SDMMC1_WP | PIOA29 | medium |
| | | | SDMMC1_CD | PIOA30 | medium |

.....continued

| NVM Bootloader | Peripheral | IO Set | Pin | PIO Line | Drive Strength (MRL C only) |
|---|---|---|---|---|---|
| NAND Flash | HSMC | 1 | D0–D7 | PIOA22-PIOA29 | low |
| | | | NANDWE | PIOA30 | low |
| | | | NANDCS3 | PIOA31 | low |
| | | | NAND ALE | PIOB0 | low |
| | | | NAND CLE | PIOB1 | low |
| | | | NANDOE | PIOB2 | low |
| | | 2 | D0–D7 | PIOA0–PIOA7 | low |
| | | | NANDWE | PIOA8 | low |
| | | | NANDCS3 | PIOA9 | low |
| | | | NAND ALE | PIOA10 | low |
| | | | NAND CLE | PIOA11 | low |
| | | | NANDOE | PIOA12 | low |
| SPI Flash | SPI_0 | 1 | SPCK | PIOA14 | low |
| | | | MOSI | PIOA15 | low |
| | | | MISO | PIOA16 | medium |
| | | | NPCS0 | PIOA17 | low |
| | | 2 | NPCS0 | PIOA30 | low |
| | | | MISO | PIOA31 | medium |
| | | | MOSI | PIOB0 | low |
| | | | SPCK | PIOB1 | low |
| | SPI_1 | 1 | SPCK | PIOC1 | low |
| | | | MOSI | PIOC2 | low |
| | | | MISO | PIOC3 | medium |
| | | | NPCS0 | PIOC4 | low |
| | | 2 | SPCK | PIOA22 | low |
| | | | MOSI | PIOA23 | low |
| | | | MISO | PIOA24 | medium |
| | | | NPCS0 | PIOA25 | low |
| | | 3 | SPCK | PIOD25 | low |
| | | | MOSI | PIOD26 | low |
| | | | MISO | PIOD27 | medium |
| | | | NPCS0 | PIOD28 | low |

.....continued

| NVM Bootloader | Peripheral | IO Set | Pin | PIO Line | Drive Strength (MRL C only) |
|---|---|---|---|---|---|
| QSPI Flash | QSPI_0 | 1 | SCK | PIOA0 | low |
| | | | CS | PIOA1 | low |
| | | | IO0 | PIOA2 | low |
| | | | IO1 | PIOA3 | low |
| | | | IO2 | PIOA4 | low |
| | | | IO3 | PIOA5 | low |
| | QSPI_0 | 2 | SCK | PIOA14 | low |
| | | | CS | PIOA15 | low |
| | | | IO0 | PIOA16 | medium |
| | | | IO1 | PIOA17 | medium |
| | | | IO2 | PIOA18 | medium |
| | | | IO3 | PIOA19 | medium |
| | QSPI_0 | 3 | SCK | PIOA22 | low |
| | | | CS | PIOA23 | low |
| | | | IO0 | PIOA24 | medium |
| | | | IO1 | PIOA25 | medium |
| | | | IO2 | PIOA26 | medium |
| | | | IO3 | PIOA27 | medium |
| | QSPI_1 | 1 | SCK | PIOA6 | low |
| | | | CS | PIOA7 | medium |
| | | | IO0 | PIOA8 | medium |
| | | | IO1 | PIOA9 | medium |
| | | | IO2 | PIOA10 | medium |
| | | | IO3 | PIOA11 | low |
| | QSPI_1 | 2 | SCK | PIOB5 | low |
| | | | CS | PIOB6 | low |
| | | | IO0 | PIOB7 | medium |
| | | | IO1 | PIOB8 | medium |
| | | | IO2 | PIOB9 | medium |
| | | | IO3 | PIOB10 | medium |
| | QSPI_1 | 3 | SCK | PIOB14 | low |
| | | | CS | PIOB15 | low |
| | | | IO0 | PIOB16 | medium |
| | | | IO1 | PIOB17 | medium |
| | | | IO2 | PIOB18 | medium |
| | | | IO3 | PIOB19 | medium |

.....continued

| NVM Bootloader | Peripheral | IO Set | Pin | PIO Line | Drive Strength (MRL C only) |
|---|---|---|---|---|---|
| Console Terminal and SAM-BA Monitor | UART_0 | 1 | DRXD | PIOB26 | low |
| | | | DTXD | PIOB27 | low |
| | UART_1 | 1 | DRXD | PIOD2 | low |
| | | | DTXD | PIOD3 | low |
| | | 2 | DRXD | PIOC7 | low |
| | | | DTXD | PIOC8 | low |
| | UART_2 | 1 | DRXD | PIOD4 | low |
| | | | DTXD | PIOD5 | low |
| | | 2 | DRXD | PIOD23 | low |
| | | | DTXD | PIOD24 | low |
| | | 3 | DRXD | PIOD19 | low |
| | | | DTXD | PIOD20 | low |
| | UART_3 | 1 | DRXD | PIOC12 | low |
| | | | DTXD | PIOC13 | low |
| | | 2 | DRXD | PIOC31 | low |
| | | | DTXD | PIOD0 | low |
| | | 3 | DRXD | PIOB11 | low |
| | | | DTXD | PIOB12 | low |
| | UART_4 | 1 | DRXD | PIOB3 | low |
| | | | DTXD | PIOB4 | low |

.....continued

| NVM Bootloader | Peripheral | IO Set | Pin | PIO Line | Drive Strength (MRL C only) |
|---|---|---|---|---|---|
| Debug Port | JTAG | 1 | TCK | PIOD14 | low |
| | | | TDI | PIOD15 | low |
| | | | TDO | PIOD16 | low |
| | | | TMS | PIOD17 | low |
| | | | NTRST | PIOD18 | low |
| | | 2 | TCK | PIOD6 | low |
| | | | TDI | PIOD7 | low |
| | | | TDO | PIOD8 | low |
| | | | TMS | PIOD9 | low |
| | | | NTRST | PIOD10 | low |
| | | 3 | TCK | PIOD27 | low |
| | | | TDI | PIOD28 | low |
| | | | TDO | PIOD29 | low |
| | | | TMS | PIOD30 | low |
| | | | NTRST | PIOD31 | low |
| | | 4 | TCK | PIOA22 | low |
| | | | TDI | PIOA23 | low |
| | | | TDO | PIOA24 | low |
| | | | TMS | PIOA25 | low |
| | | | NTRST | PIOA26 | low |

## 16.6 SAM-BA Monitor

This part of the ROM code is executed when no valid code is found in any NVM during the NVM boot sequence, and if the DISABLE_MONITOR Fuse bit is not set.

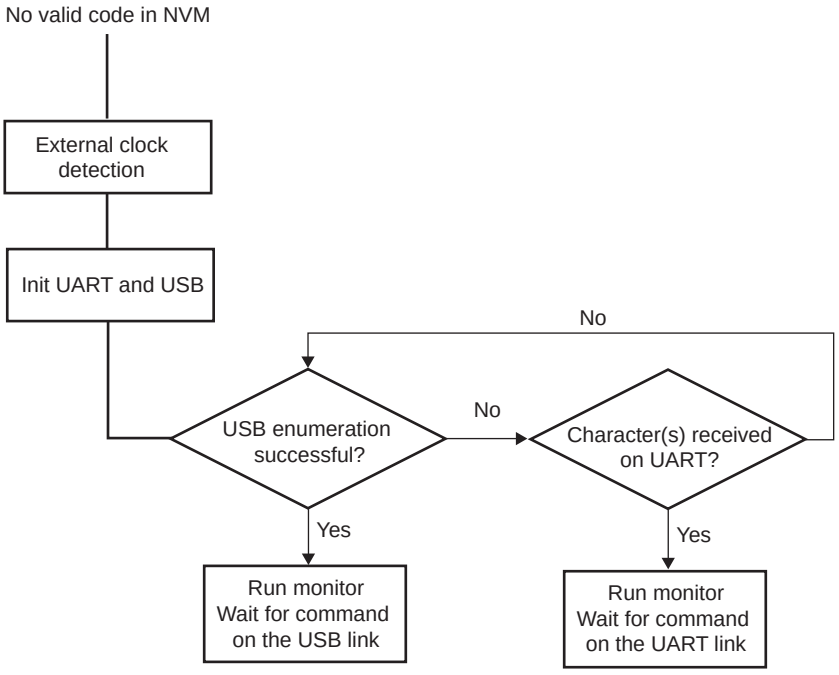
The Main Oscillator is enabled and set in Bypass mode. If the MOSCSELS bit rises, an external clock is connected. If not, the Bypass mode is cleared to attempt external quartz detection. This detection is successful when the MOSCXTS and MOSCSELS bits rise, else the internal 12 MHz fast RC oscillator is used as the Main Clock.

If an external clock or crystal frequency is found, then the PLLA is configured to allow communication on the USB link for the SAM-BA Monitor, else the Main Clock is switched back to the internal 12 MHz fast RC oscillator and USB is not activated. The SAM-BA Monitor steps are:

- Initialize UART and USB.
- Check if USB Device enumeration occurred.
- Check if characters are received on the UART.

Once the communication interface is identified, the application runs in an infinite loop waiting for different commands as listed in the table "Commands Available through the SAM-BA Monitor".

**Figure 16-13. SAM-BA Monitor**




### 16.6.1 Command List

**Table 16-6. Commands Available through the SAM-BA Monitor**

| Command | Action | Argument(s) | Example |
|---|---|---|---|
| N | Set Normal Mode | No argument | N# |
| T | Set Terminal Mode | No argument | T# |
| O | Write a byte | Address, Value# | O200001,CA# |
| o | Read a byte | Address,# | o200001,# |
| H | Write a half word | Address, Value# | H200002,CAFE# |
| h | Read a half word | Address,# | h200002,# |
| W | Write a word | Address, Value# | W200000,CAFEDECA# |
| w | Read a word | Address,# | w200000,# |
| S | Send a file | Address,# | S200000,# |
| R | Receive a file | Address, NbOfBytes# | R200000,1234# |
| G | Go | Address# | G200200# |
| V | Display version | No argument | V# |

- Mode commands:
  - Normal mode configures SAM-BA Monitor to send / receive data in binary format,

  - Terminal mode configures SAM-BA Monitor to send / receive data in ASCII format.
- Write commands: Writes a byte (O), a halfword (H) or a word (W) to the target
  - Address: Address in hexadecimal
  - Value: Byte, halfword or word to write in hexadecimal
  - Output: '>'
- Read commands: Reads a byte (o), a halfword (h) or a word (w) from the target
  - Address: Address in hexadecimal
  - Output: The byte, halfword or word read in hexadecimal followed by '>'
- Send a file (S): Sends a file to a specified address
  - Address: Address in hexadecimal
  - Output: '>'
    **Note:** There is a timeout on this command which is reached when the prompt '>' appears before the end of the command execution.
- Receive a file (R): Receives data into a file from a specified address
  - Address: Address in hexadecimal
  - NbOfBytes: Number of bytes in hexadecimal to receive
  - Output: '>'
- Go (G): Jumps to a specified address and executes the code
  - Address: Address to jump to in hexadecimal
  - Output: '>' once returned from the program execution. If the executed program does not handle the link register at its entry and does not return, the prompt is not displayed.
- Get Version (V): Returns the Boot Program version
  - Output: version, date and time of ROM code followed by '>'

### 16.6.2 UART Port

Communication is performed through the UART port initialized to 115,200 bauds, 8 bits of data, no parity, 1 stop bit.

### 16.6.3 Xmodem Protocol

The Send and Receive File commands use the Xmodem protocol to communicate. Any terminal using this protocol can be used to send the application file to the target. The size of the binary file to send depends on the SRAM size embedded in the product. In all cases, the size of the binary file must be lower than the SRAM size because the Xmodem protocol requires some SRAM memory in order to work.

The Xmodem protocol supported is the 128-byte length block. This protocol uses a two-character CRC16 to guarantee detection of maximum bit errors.

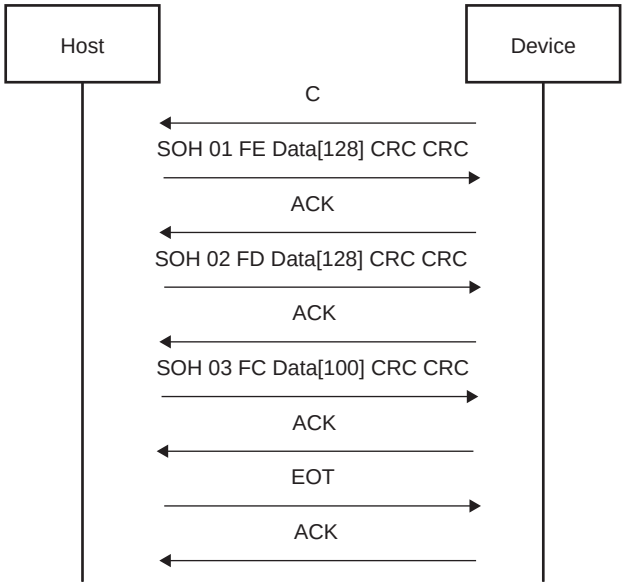
The Xmodem protocol with CRC is supported by successful transmission reports provided both by a sender and by a receiver. Each transfer block is as follows:

<SOH><blk #><255-blk #><--128 data bytes--><checksum> in which:

- <SOH> = 01 hex
- <blk #> = binary number, starts at 01, increments by 1, and wraps 0FFH to 00H (not to 01)
- <255-blk #> = 1's complement of the blk#.
- <checksum> = 2 bytes CRC16

The figure below shows a transmission using this protocol.

**Figure 16-14. Xmodem Transfer Example**




### 16.6.4 USB Device Port

#### 16.6.4.1 Supported External Crystal / External Clocks

The SAM-BA Monitor only supports an external crystal or external clock frequency at 12 MHz to allow USB communication.

#### 16.6.4.2 Supported External Crystal / External Clocks

The device uses the USB Communication Device Class (CDC) drivers to take advantage of the installed PC Serial Communication software to talk over the USB. The CDC is implemented in all releases of Windows®, starting from Windows 98SE®. The CDC document, available at www.usb.org, describes how to implement devices such as ISDN modems and virtual COM ports.

Vendor ID is 0x03EB. The product ID is 0x6124. These references are used by the host operating system to mount the correct driver. On Windows systems, INF files contain the correspondence between vendor ID and product ID.

#### 16.6.4.3 Supported External Crystal / External Clocks

The USB protocol is a master/slave protocol. The host starts the enumeration, sending requests to the device through the control endpoint. The device handles standard requests as defined in the USB Specification.

**Table 16-7. Handled Standard Requests**

| Request | Definition |
|---|---|
| GET_DESCRIPTOR | Returns the current device configuration value |
| SET_ADDRESS | Sets the device address for all future device access |
| SET_CONFIGURATION | Sets the device configuration |
| GET_CONFIGURATION | Returns the current device configuration value |
| GET_STATUS | Returns status for the specified recipient |
| SET_FEATURE | Used to set or enable a specific feature |

| .....continued | |
|---|---|
| **Request** | **Definition** |
| CLEAR_FEATURE | Used to clear or disable a specific feature |

The device also handles some class requests defined in the CDC class.

**Table 16-8. Handled Class Requests**

| Request | Definition |
|---|---|
| SET_LINE_CODING | Configures DTE rate, stop bits, parity and number of character bits |
| GET_LINE_CODING | Requests current DTE rate, stop bits, parity and number of character bits |
| SET_CONTROL_LINE_STATE | RS-232 signal used to indicate to the DCE device that the DTE device is now present |

Unhandled requests are stalled.

#### 16.6.4.4 Communication Endpoints

Endpoint 0 is used for the enumeration process.

Endpoint 1 (64-byte Bulk OUT) and endpoint 2 (64-byte Bulk IN) are used as communication endpoints.

SAM-BA Boot commands are sent by the host through Endpoint 1. If required, the message is split into several data payloads by the host driver.

If the command requires a response, the host sends IN transactions to pick up the response.

## 16.7 Fuse Box Controller

Read/write access to the fuse bits requires that the internal 12 MHz RC oscillator is enabled.

### 16.7.1 Fuse Bit Mapping

One 32-bit word is reserved for boot configuration.

512 fuse bits are available for customer needs.

Writing a ‘1’ to SFR_SECURE.FUSE disables access to the Secure Fuse Controller (SFC).

To avoid any malfunctioning, the user must not write the “DO NOT USE (DNU)” fuse bits in the Boot Configuration area.

**Table 16-9. Customer Fuse Matrix**

| SFC_DR | Bits | Use | | |
|---|---|---|---|---|
| 16 | [543:512] | JTAG_DIS[543] | SEC_DEBUG_DIS[542] | Boot Configuration bits[541:512](1) |

.....continued

| SFC_DR | Bits | Use |
|---|---|---|
| 15 | [511:480] | USER_DATA[511:0] |
| 14 | [479:448] | |
| 13 | [447:416] | |
| 12 | [415:384] | |
| 11 | [383:352] | |
| 10 | [351:320] | |
| 9 | [319:288] | |
| 8 | [287:256] | |
| 7 | [255:224] | |
| 6 | [223:192] | |
| 5 | [191:160] | |
| 4 | [159:128] | |
| 3 | [127:96] | |
| 2 | [95:64] | |
| 1 | [63:32] | |
| 0 | [31:0] | |

**Note:** See section Boot Configuration Word for details on the contents of these bits.

**Table 16-10. Special Function Bits**

| JTAG Disable (Fuse bit 543) | Secure Debug Disable (Fuse bit 542) | Description |
|---|---|---|
| 0 | 0 | Full JTAG debug allowed in Secure and Normal modes |
| 0 | 1 | JTAG debug allowed in Normal mode only (not in Secure mode) |
| 1 | X | JTAG debug disabled |

# 17. AXI Matrix (AXIMX)

## 17.1 Description

The AXI Matrix comprises the embedded Advanced Extensible Interface (AXI) bus protocol which supports separate address/control and data phases, unaligned data transfers using byte strobes, burst-based transactions with only start address issued, separate read and write data channels to enable low-cost DMA, ability to issue multiple outstanding addresses, out-of-order transaction completion, and easy addition of register stages to provide timing closure.

## 17.2 Embedded Characteristics

- High-Performance AXI Network Interconnect
- One Master
  - Cortex-A5 Core
- Two Slaves
  - ROM
  - AXI/AHB bridge to AHB Matrix
- Single-Cycle Arbitration
- Full Pipelining to Prevent Master Stalls
- One Remap State

## 17.3 Operation

### 17.3.1 Remap

Remap states are managed in the AXI Matrix Remap register (AXIMX_REMAP): AXIMX_REMAP.REMAP0 (register bit 0) is used to remap RAM @ addr 0x00000000.

The number of remap states can be defined using eight bits of the AXIMX_REMAP register, and a bit in AXIMX_REMAP controls each remap state.

Each remap state can be used to control the address decoding for one or more slave interfaces. If a slave interface is affected by two remap states that are both asserted, the remap state with the lowest remap bit number takes precedence.

Each slave interface can be configured independently so that a remap state can perform different functions for different masters.

A remap state can:

- Alias a memory region into two different address ranges
- Move an address region
- Remove an address region

Because of the nature of the distributed register subsystem, the masters receive the updated remap bit states in sequence, and not simultaneously.

A slave interface does not update to the latest remap bit setting until:

- The address completion handshake accepts any transaction that is pending
- Any current lock sequence completes

At powerup, ROM is seen at address 0. After powerup, the internal SRAM can be moved down to address 0 by means of the remap bits.

## 17.4 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | AXIMX_REMAP | 7:0 | | | | | | | | REMAP0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 17.4.1 AXI Matrix Remap Register

**Name:** AXIMX_REMAP
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | REMAP0 |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – REMAP0** Remap State 0
SRAM is seen at address 0x00000000 (through AHB slave interface) instead of ROM.

# 18. Matrix (H64MX/H32MX)

## 18.1 Description

The system embeds three system bus matrixes: one based on the AXI protocol (AXIMX) and two based on the AHB protocol (H64MX and H32MX). This section describes the implementation of the 64-bit Matrix (H64MX) and the 32-bit Matrix (H32MX).

For details on the AXIMX matrix, refer to the section “AXI Matrix (AXIMX)”.

Each matrix implements a multilayer system bus, which enables parallel access paths between multiple masters and slaves in a system, thus increasing the overall bandwidth. The normal latency to connect a master to a slave is one cycle, except for the default master of the accessed slave which is connected directly (zero cycle latency).

**Note:** When a master and a slave are on different bus matrixes (AXIMX, H64MX, or H32MX), both matrixes (H64MX and H32MX) and the bridge between the bus matrixes must be configured accordingly.

## 18.2 Embedded Characteristics

- 32-bit or 64-bit Data Bus
- 64-bit Matrix (H64MX) Providing 12 Masters and 15 Slaves
- 32-bit Matrix (H32MX) Providing 8 Masters and 6 Slaves
- One Address Decoder for Each Master
- Support for Long Bursts of Length 32, 64, 128 and Up to the Limit of 256-bit Burst Beats of Words
- Enhanced Programmable Mixed Arbitration for Each Slave:
  - Round-robin
  - Fixed priority
  - Latency quality of service
- Programmable Default Master for Each Slave:
  - No default master
  - Last accessed default master
  - Fixed default master
- Deterministic Maximum Access Latency for Masters
- Zero or One Cycle Arbitration Latency for the First Access of a Burst
- Bus Lock Forwarding to Slaves
- One Special Function Register for Each Slave (not dedicated)
- Register Write Protection
- ARM TrustZone Technology

## 18.3 64-bit Matrix (H64MX)

### 18.3.1 Matrix Masters

The H64MX manages 12 masters, which means that each master can perform an access, concurrently with others, to an available slave.

This matrix operates at MCK.

Each master has its own decoder, which is defined specifically for each master. In order to simplify the addressing, all the masters have the same decodings.

**Table 18-1. List of H64MX Masters**

| Master No. | Name | Security Type |
|---|---|---|
| 0 | Bridge from AXI Matrix (Core) | Not applicable |
| 1, 2 | DMA Controller 0 | Peripheral Securable |
| 3, 4 | DMA Controller 1 | Peripheral Securable |
| 5, 6 | LCDC DMA | Peripheral Securable |
| 7 | SDMMC0 | Peripheral Securable |
| 8 | SDMMC1 | Peripheral Securable |
| 9 | ISC DMA | Peripheral Securable |
| 10 | AESB | Not applicable(1) |
| 11 | Bridge from H32MX to H64MX | Not applicable |

**Note:**

1. Master signals secure/not secure are propagated through the AES bridge.

### 18.3.2 Matrix Slaves

The H64MX manages 15 slaves. Each slave has its own arbiter providing a dedicated arbitration per slave.

**Table 18-2. List of H64MX Slaves**

| Slave No. | Description | TrustZone Access Management |
|---|---|---|
| 0 | Bridge from H64MX to H32MX | Not applicable |
| 1 | H64MX Peripheral Bridge | HSEL0: not applicable |
| | SDMMC0 | HSEL1: Internal Securable to Peripheral: 1 region(1) |
| | SDMMC1 | HSEL2: Internal Securable to Peripheral: 1 region(1) |
| 2 | DDR2 Port 0 - AESB | Scalable Securable: 4 regions(2) |
| 3 | DDR2 Port 1 | Scalable Securable: 4 regions(2) |
| 4 | DDR2 Port 2 | Scalable Securable: 4 regions(2) |
| 5 | DDR2 Port 3 | Scalable Securable: 4 regions(2) |
| 6 | DDR2 Port 4 | Scalable Securable: 4 regions(2) |
| 7 | DDR2 Port 5 | Scalable Securable: 4 regions(2) |
| 8 | DDR2 Port 6 | Scalable Securable: 4 regions(2) |
| 9 | DDR2 Port 7 | Scalable Securable: 4 regions(2) |

| .....continued | | |
|---|---|---|
| **Slave No.** | **Description** | **TrustZone Access Management** |
| 10 | Internal SRAM 128K | Internal Securable: 1 region |
| 11 | Internal SRAM 128K (Cache L2) | Internal Securable: 1 region |
| 12 | QSPI0 | Internal Securable: 1 region |
| 13 | QSPI1 | Internal Securable: 1 region |
| 14 | AESB | Not applicable |

**Note:**

1. For each SDMMCx, see "Security Types of SDMMC System Bus Slaves" for Internal Securable to Peripheral type configuration. A consistent configuration must be done for:
   - the slave port,
   - MATRIX_SPSELSR for the general interrupt and the master port,
   - MATRIX_SPSELSR for the TIMER interrupt.
2. For consistency, each DDR2 port must have the same TrustZone access management configuration.

### 18.3.3 Master to Slave Access

The following table shows how masters and slaves interconnect. Writing in a register or field not dedicated to a master or a slave has no effect.

**Table 18-3. Master to Slave Access on H64MX**

| | | MASTER | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| **SLAVE** | | **Bridge from AXIMX (Core)** | **XDMAC0** | | **XDMAC1** | | **LCDC DMA** | | **SDMMC0 DMA** | **SDMMC1 DMA** | **ISC DMA** | **AESB** | **Bridge from H32MX** |
| 0 | Bridge from H64MX to H32MX | X | X | X | X | X | – | – | – | – | – | – | – |
| 1 | H64MX Peripheral Bridge | X | X | X | X | X | – | – | – | – | – | – | X |
| | SDMMC0–SDMMC1 | X | X | X | X | X | – | – | – | – | – | – | X |
| 2 | DDR2 Port 0 | – | – | – | – | – | – | – | – | – | – | X(1) | – |
| 3 | DDR2 Port 1 | X | – | – | – | – | – | – | – | – | – | – | – |
| 4 | DDR2 Port 2 | – | – | – | – | – | X | – | – | – | – | – | – |
| 5 | DDR2 Port 3 | – | – | – | – | – | – | X | – | – | – | – | – |
| 6 | DDR2 Port 4 | – | – | – | – | – | – | – | X | X | X | – | – |

.....continued

| | | MASTER | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| SLAVE | | Bridge from AXIMX (Core) | XDMAC0 | | XDMAC1 | | LCDC DMA | | SDMMC0 DMA | SDMMC1 DMA | ISC DMA | AESB | Bridge from H32MX |
| 7 | DDR2 Port 5 | – | X | – | X | – | – | – | – | – | – | – | – |
| 8 | DDR2 Port 6 | – | – | X | – | X | – | – | – | – | – | – | – |
| 9 | DDR2 Port 7 | – | – | – | – | – | – | – | – | – | – | – | X |
| 10 | Internal SRAM | X | X | X | X | X | X | X | X | X | X | – | X |
| 11 | L2C SRAM | X | X | X | X | X | X | X | X | X | X | – | X |
| 12 | QSPI0 | X | X | X | X | X | – | – | – | – | – | X(1) | X |
| 13 | QSPI1 | X | X | X | X | X | – | – | – | – | – | X(1) | X |
| 14 | AESB | X | X | X | X | X | – | – | – | – | – | – | X |

**Note:**

1. To avoid deadlock when accessing the AESB slave, the QSPI0, QSPI1 and DDR2 Port 0 Slave Configuration registers (MATRIX_SCFGx) must be configured either with DEFMSTR_TYPE = NONE ('0') or with DEFMSTR_TYPE = FIXED ('2') and FIXED_DEFMSTR = 10.

## 18.4 32-bit Matrix (H32MX)

### 18.4.1 Matrix Masters

The H32MX manages eight masters, which means that each master can perform an access, concurrently with others, to an available slave.

This matrix can operate at MCK if MCK is lower than 83 MHz, or at MCK/2 if MCK is higher than 83 MHz. Refer to the section “Power Management Controller (PMC)” for more details.

Each master has its own decoder, which is defined specifically for each master. In order to simplify the addressing, all the masters have the same decodings.

**Table 18-4. List of H32MX Masters**

| Master No. | Name | Security Type |
|---|---|---|
| 0 | Bridge from H64MX to H32MX | Not applicable |
| 1 | Integrity Check Monitor (ICM) | Peripheral Securable |
| 2 | UHPHS EHCI DMA | Peripheral Securable |
| 3 | UHPHS OHCI DMA | Peripheral Securable |
| 4 | UDPHS DMA | Peripheral Securable |
| 5 | GMAC DMA | Peripheral Securable |

| .....continued | | |
|---|---|---|
| **Master No.** | **Name** | **Security Type** |
| 6 | CAN0 DMA | Peripheral Securable |
| 7 | CAN1 DMA | Peripheral Securable |

### 18.4.2 Matrix Slaves

The H32MX manages six slaves. Each slave has its own arbiter providing a dedicated arbitration per slave.

**Table 18-5. List of H32MX Slaves**

| Slave No. | Description | TZ Access Management |
|---|---|---|
| 0 | Bridge from H32MX to H64MX | Not applicable |
| 1 | H32MX Peripheral Bridge 0 | Not applicable |
| 2 | H32MX Peripheral Bridge 1 | Not applicable |
| 3 | External Bus Interface | External Securable: 7 regions:<br>HSEL0: 0x10000000 128 MB CS0<br>HSEL1: 0x18000000 128 MB CS0<br>HSEL2: 0x60000000 128 MB CS1<br>HSEL3: 0x68000000 128 MB CS1<br>HSEL4: 0x70000000 128 MB CS2<br>HSEL5: 0x78000000 128 MB CS2<br>HSEL6: 0x80000000 128 MB CS3 |
| | NFC Command Register | Internal Securable to Peripheral: 1 region<br>HSEL7: 0xC0000000 256 MB NFCCMD |
| 4 | NFC SRAM | Internal Securable: 1 region |
| 5 | USB Device High Speed (UDPHS) Dual Port RAM (DPR) | HSEL0: Internal Securable: 1 region |
| | USB Host (UHPHS) OHCI registers | HSEL1: Internal Securable to Peripheral: 1 region(Note) |
| | USB Host (UHPHS) EHCI registers | HSEL2: Internal Securable to Peripheral: 1 region(Note) |
| 6 | Peripheral Touch Controller (PTC) | Internal Securable: 1 region |

**Note:** UHPHS: Consistent configuration must be done on:

- Slave UHPHS OHCI Internal Securable Peripheral,
- Slave UHPHS EHCI Internal Securable Peripheral,
- MATRIX_SPSELSR for Interrupt and Master

### 18.4.3 Master to Slave Access

The following table shows how masters and slaves interconnect. Writing in a register or field not dedicated to a master or a slave has no effect.

**Table 18-6. Master to Slave Access on H32MX**

| | | MASTER | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0 (Through Bridge from H64MX) | | | | | 1 | 2 | 3 | 4 | 5 | 6 |
| SLAVE | | Core | XDMAC0 | | XDMAC1 | | ICM | UHPHS EHCI DMA | UHPHS OHCI DMA | UDPHS DMA | GMAC DMA | CAN0 DMA |
| | | | IF0 | IF1 | IF0 | IF1 | | | | | | |
| 0 | Bridge from H32MX to H64MX | – | – | – | – | – | X | X | X | X | X | X |
| 1 | H32MX Peripheral Bridge 0 | X | – | X | – | X | – | – | – | – | – | – |
| 2 | H32MX Peripheral Bridge 1 | X | – | X | – | X | – | – | – | – | – | – |
| 3 | EBI CS0..CS3 | X | X | – | X | – | X | – | – | – | – | – |
| | NFC Command Register | X | X | – | X | – | – | – | – | – | – | – |
| 4 | NFC SRAM | X | X | – | X | – | – | – | – | – | – | – |
| 5 | UDPHS RAM | X | – | – | – | X | – | – | – | – | – | – |
| | UHP OHCI Reg | X | – | – | – | X | – | – | – | – | – | – |
| | UHP EHCI Reg | X | – | – | – | X | – | – | – | – | – | – |
| 6 | Peripheral Touch Controller (PTC) | X | – | – | – | – | – | – | – | – | – | – |

## 18.5 Memory Mapping

The MATRIX provides one decoder for every master interface. The decoder offers each master several memory mappings. Each memory area can be assigned to several slaves. Booting at the same address while using different slaves (i.e., external RAM, internal ROM or internal Flash, etc.) becomes possible.

## 18.6 Special Bus Granting Mechanism

The MATRIX provides some speculative bus granting techniques in order to anticipate access requests from masters. This mechanism reduces latency at first access of a burst, or for a single transfer, as long as the slave is free from any other master access. It does not provide any benefit if the slave is

continuously accessed by more than one master, since arbitration is pipelined and has no negative effect on the slave bandwidth or access latency.

This bus granting mechanism sets a different default master for every slave.

At the end of the current access, if no other request is pending, the slave remains connected to its associated default master. A slave can be associated with three kinds of default masters:

- No default master
- Last access master
- Fixed default master

To change from one type of default master to another, the user interface provides Slave Configuration registers (MATRIX_SCFGx), one for every slave, which set a default master for each slave. MATRIX_SCFGx contains two fields to manage master selection: DEFMSTR_TYPE and FIXED_DEFMSTR. The 2-bit DEFMSTR_TYPE field selects the default master type (no default, last access master, fixed default master), whereas the 4-bit FIXED_DEFMSTR field selects a fixed default master provided that DEFMSTR_TYPE is set to fixed default master. See “Bus Matrix Slave Configuration Registers”.

## 18.7 No Default Master

After the end of the current access, if no other request is pending, the slave is disconnected from all masters.

This configuration incurs one latency clock cycle for the first access of a burst after bus Idle. Arbitration without default master can be used for masters that perform significant bursts or several transfers with no Idle in between, or if the slave bus bandwidth is widely used by one or more masters.

This configuration provides no benefit on access latency or bandwidth when reaching maximum slave bus throughput regardless of the number of requesting masters.

## 18.8 Last Access Master

After the end of the current access, if no other request is pending, the slave remains connected to the last master that performed an access request.

This allows the MATRIX to remove the one latency cycle for the last master that accessed the slave. Other nonprivileged masters still get one latency clock cycle if they need to access the same slave. This technique is used for masters that mainly perform single accesses or short bursts with some Idle cycles in between.

This configuration provides no benefit on access latency or bandwidth when reaching maximum slave bus throughput whatever is the number of requesting masters.

## 18.9 Fixed Default Master

After the end of the current access, if no other request is pending, the slave connects to its fixed default master. Unlike the last access master, the fixed default master does not change unless the user modifies it by software (FIXED_DEFMSTR field of the related MATRIX_SCFG).

This allows the MATRIX arbiters to remove the one latency clock cycle for the fixed default master of the slave. All requests attempted by the fixed default master do not cause any arbitration latency, whereas

other nonprivileged masters will get one latency cycle. This technique is used for a master that mainly performs single accesses or short bursts with Idle cycles in between.

This configuration provides no benefit on access latency or bandwidth when reaching maximum slave bus throughput, regardless of the number of requesting masters.

## 18.10 Arbitration

The MATRIX provides an arbitration mechanism that reduces latency when conflicts occur, i.e., when two or more masters try to access the same slave at the same time. One arbiter per slave is provided, thus arbitrating each slave specifically.

The user can choose between two arbitration types or mix them for each slave:

- Round-robin Arbitration (default)
- Fixed Priority Arbitration

The resulting algorithm may be complemented by selecting a default master configuration for each slave.

When rearbitration must be done, specific conditions apply. See “Arbitration Scheduling” .

### 18.10.1 Arbitration Scheduling

Each arbiter has the ability to arbitrate between two or more master requests. In order to avoid burst breaking and also to provide the maximum throughput for slave interfaces, arbitration takes place during the following cycles:

- Idle Cycles: when a slave is not connected to any master or is connected to a master which is not currently accessing it.
- Single Cycles: when a slave is currently performing a single access.
- End of Burst Cycles: when the current cycle is the last cycle of a burst transfer. For defined burst length, predicted end of burst matches the size of the transfer but is managed differently for undefined burst length. See “Undefined Length Burst Arbitration” .
- Slot Cycle Limit: when the slot cycle counter has reached the limit value indicating that the current master access is too long and must be broken. See “Slot Cycle Limit Arbitration” .

#### 18.10.1.1 Undefined Length Burst Arbitration

To prevent long burst lengths that can lock the access to the slave for an excessive period of time, the user can trigger the rearbitration before the end of the incremental bursts. The rearbitration period can be selected from the following Undefined Length Burst Type (ULBT) possibilities:

- Unlimited: no predetermined end of burst is generated. This value enables 1 Kbyte burst lengths.
- 1-beat bursts: predetermined end of burst is generated at each single transfer during the INCR transfer.
- 4-beat bursts: predetermined end of burst is generated at the end of each 4-beat boundary during INCR transfer.
- 8-beat bursts: predetermined end of burst is generated at the end of each 8-beat boundary during INCR transfer.
- 16-beat bursts: predetermined end of burst is generated at the end of each 16-beat boundary during INCR transfer.
- 32-beat bursts: predetermined end of burst is generated at the end of each 32-beat boundary during INCR transfer.

© 2018 Microchip Technology Inc.

- 64-beat bursts: predetermined end of burst is generated at the end of each 64-beat boundary during INCR transfer.
- 128-beat bursts: predetermined end of burst is generated at the end of each 128-beat boundary during INCR transfer.

Undefined-length bursts lower than 8 beats should not be used since this may decrease the overall bus bandwidth due to arbitration and slave latencies at each first access of a burst.

However, if the length of undefined-length bursts is known for a master, it is recommended to configure MATRIX_MCFG.ULBT accordingly.

#### 18.10.1.2 Slot Cycle Limit Arbitration

The MATRIX contains specific logic to break long accesses, such as very long bursts on a very slow slave (e.g., an external low speed memory). At each arbitration time, a counter is loaded with the value previously written in the SLOT_CYCLE field of the related Slave Configuration Register (MATRIX_SCFG) and decreased at each clock cycle. When the counter elapses, the arbiter has the ability to rearbitrate at the end of the current system bus access cycle.

Unless a master has a very tight access latency constraint, which could lead to data overflow or underflow due to a badly undersized internal FIFO with respect to its throughput, the Slot Cycle Limit should be disabled (SLOT_CYCLE = 0) or set to its default maximum value in order not to inefficiently break long bursts performed by some masters.

In most cases, this feature is not needed and should be disabled for power saving.

**⚠WARNING** This feature cannot prevent any slave from locking its access indefinitely.

### 18.10.2 Arbitration Priority Scheme

The MATRIX arbitration scheme is organized in priority pools, each corresponding to an access criticality class as shown in the “Latency Quality of Service” column in the following table.

**Table 18-7. Arbitration Priority Pools**

| Priority Pool | Latency Quality of Service |
|---|---|
| 3 | Latency Critical |
| 2 | Latency Sensitive |
| 1 | Bandwidth Sensitive |
| 0 | Background Transfers |

Round-robin priority is used in the highest and lowest priority pools 3 and 0, whereas fixed level priority is used between priority pools and in the intermediate priority pools 2 and 1. See “Round-robin Arbitration” .

For each slave, each master is assigned to one of the slave priority pools through the priority registers for slaves (MxPR fields of MATRIX_PRAS and MATRIX_PRBS). When evaluating master requests, this priority pool level always takes precedence.

After reset, most of the masters belong to the lowest priority pool (MxPR = 0, Background Transfer) and are therefore granted bus access in a true round-robin order.

The highest priority pool must be specifically reserved for masters requiring very low access latency. If more than one master belongs to this pool, they will be granted bus access in a biased round-robin manner which allows tight and deterministic maximum access latency from system bus requests. In the worst case, any currently occurring high-priority master request will be granted after the current bus master access has ended and other high priority pool master requests, if any, have been granted once each.

The lowest priority pool shares the remaining bus bandwidth between masters.

Intermediate priority pools allow fine priority tuning. Typically, a latency-sensitive master or a bandwidth-sensitive master will use such a priority level. The higher the priority level (MxPR value), the higher the master priority.

To optimize processor performance, it is recommended configure CPU priority with the default reset value 2 (Latency Sensitive).

All combinations of MxPR values are allowed for all masters and slaves. For example, some masters might be assigned the highest priority pool (round-robin), and remaining masters the lowest priority pool (round-robin), with no master for intermediate fixed priority levels.

#### 18.10.2.1 Fixed Priority Arbitration

Fixed priority arbitration algorithm is the first and only arbitration algorithm applied between masters from distinct priority pools. It is also used in priority pools other than the highest and lowest priority pools (intermediate priority pools).

Fixed priority arbitration allows the MATRIX arbiters to dispatch the requests from different masters to the same slave by using the fixed priority defined by the user in the MxPR field for each master in the registers MATRIX_PRAS and MATRIX_PRBS. If two or more master requests are active at the same time, the master with the highest priority MxPR number is serviced first.

In intermediate priority pools, if two or more master requests with the same priority are active at the same time, the master with the highest number is serviced first.

#### 18.10.2.2 Round-robin Arbitration

This algorithm is only used in the highest and lowest priority pools. It allows the MATRIX arbiters to properly dispatch requests from different masters to the same slave. If two or more master requests are active at the same time in the priority pool, they are serviced in a round-robin increasing master number order.

## 18.11 Register Write Protection

To prevent any single software error from corrupting MATRIX behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the Write Protection Mode Register (MATRIX_WPMR).

If a write access to a write-protected register is detected, the WPVS bit in the Write Protection Status Register (MATRIX_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS flag is reset by writing the Write Protect Mode Register (MATRIX_WPMR) with the appropriate access key WPKEY.

The following registers can be write-protected:

- Bus Matrix Master Configuration Registers

- Bus Matrix Slave Configuration Registers
- Bus Matrix Priority Registers A For Slaves
- Bus Matrix Priority Registers B For Slaves
- Master Error Interrupt Enable Register
- Master Error Interrupt Disable Register
- Security Slave Registers
- Security Areas Split Slave Registers
- Security Region Top Slave Registers
- Security Peripheral Select x Registers

## 18.12 TrustZone Technology

TrustZone secure software is supported through the filtering of each slave access with master security bit extension signals.

TrustZone technology adds the ability to manage the access rights for secure and non-secure accesses. The access rights are defined through the hardware and software configuration of the device. The operating mode is as follows:

- Masters transmit requests with the secure or non-secure Security option.
- The MATRIX, according to its configuration and the request, grants or denies the access.

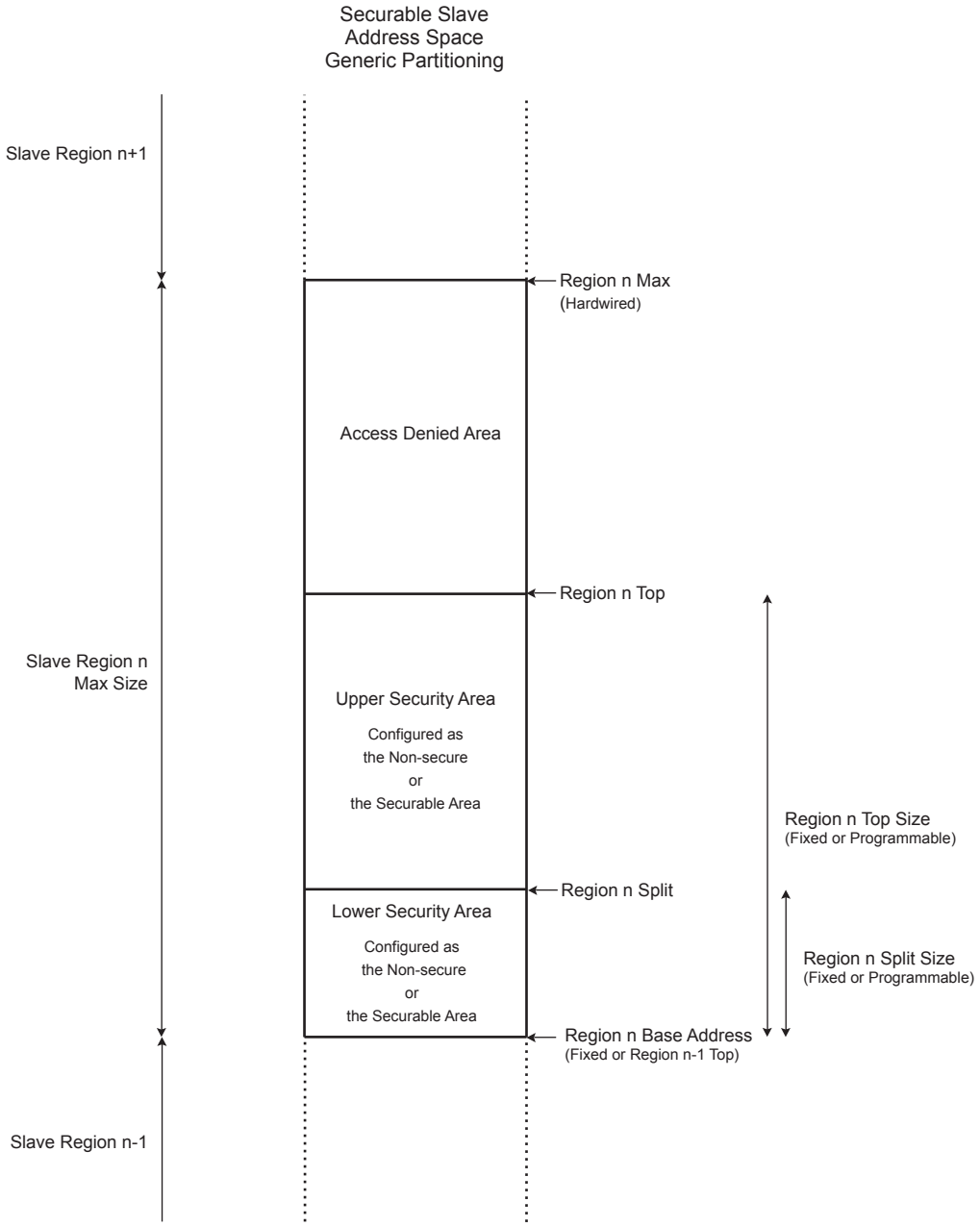
The slave address space is divided into one or more slave regions. The slave regions are generally contiguous parts of the slave address space. The slave region is potentially split into an access denied area (upper part) and a security region which can be split (lower part), unless the slave security region occupies the whole slave region. The security region itself can be split into one secure area and one non-secure area. The secure area may be independently secure for read access and for write access.

For one slave region, the following characteristics are configured by hardware or software:

- Base Address of the slave region
- Max size of the slave region: the maximum size for the region’s physical content
- Top size of the slave security region: the actually programmed or fixed size for the region’s physical content
- Split size of the slave security region: the size of the lower security area of the region.

The following figure shows how the terms defined here are implemented in a slave address space.

**Figure 18-1. Generic Partitioning of the AHB Slave Address Space**




A set of security registers allows to specify, for each slave, the slave security region or slave security area, the security mode required to access this slave, slave security region or slave security area.

Additional Bus Matrix security registers allow to specify, for each peripheral bus slave, the security mode required to access this slave (see "Security Peripheral Select x Registers").

See "Security Slave Registers".

These registers can only be accessed in Secure mode.

The MATRIX propagates the security bit down to the slaves to let them perform additional security checks, and the MATRIX itself allows or denies the access to the slaves by means of its TrustZone embedded controller.

Access violations may be reported either by a slave through the bus error response (example from the AHB/APB Bridge), or by the Bus Matrix embedded TrustZone controller. In both cases, a bus error response is sent to the offending master and the error is flagged in the Master Error Status Register. An interrupt can be sent to the Secure world, if it has been enabled for that master by writing into the Master Error Interrupt Enable Register. Thus, the offending master is identified. The offending address is registered in the Master Error Address Registers, so that the slave and the targeted security region are also known.

Depending on the hardware parameters and software configuration, the address space of each slave security region may or may not be split into two parts, one belonging to the Secure world and the other one to the Normal world.

Five different security types of slaves are supported. The number of security regions is set by design for each slave, independently, from 1 to 8, totalling from 1 up to 16 security areas for security configurable slaves.

### 18.12.1 Security Types of Slaves

#### 18.12.1.1 Principles

The MATRIX supports five different security types of slaves: two fixed types and three configurable types. The security type of a slave is set at hardware design among the following:

- Never Secure
- Always Secure
- Internal Securable
- External Securable
- Scalable Securable

The security type is set at hardware design on a per-master and a per-slave basis. **Never Secure** and **Always Secure** security types are not software configurable.

The different security types have the following characteristics:

- **Never Secure** slaves have no security mode access restriction. Their address space is precisely set by design. Any out-of-address range access is denied and reported.
- **Always Secure** slaves can only be accessed by a secure master request. Their address space is precisely set by design. Any non-secure or out-of-address range access is denied and reported.
- **Internal Securable** is intended for internal RAM. The Internal Securable slave has one slave region which has a hardware fixed base address and Security Region Top. This slave region may be split through software configuration into one Non-secure area plus one Secure area. Inside the slave security region, the split boundary is programmable in powers of 2 from 4 Kbytes up to the full slave security region address space. The security area located below the split boundary may be configured as the Non-secure or the Secure one. The Securable area may be independently configured as Read Secured and/or Write Secured. Any access with security or address range violation is denied and reported.
- **External Securable** is intended for external memories on the EBI, such as DDR, SDRAM, external ROM or NAND Flash. The External Securable slave has identical features as the Internal Securable slave, plus the ability to configure each of its slave security region address space sizes according to the external memory parts used. This avoids mirroring Secure areas into Non-secure areas, and further restricts the overall accessible address range. Any access with security or configured address range violation is denied and reported.

- **Scalable Securable** is intended for external memories with a dedicated slave, such as DDR. The Scalable Securable slave is divided into a fixed number of scalable, equally sized, and contiguous security regions. Each of them can be split in the same way as for Internal or External Securable slaves. The security region size must be configured by software, so that the equally-sized regions fill the actual available memory. This avoids mirroring Secure areas into Non-secure areas, and further restricts the overall accessible address range. Any access with security or configured address range violation is denied and reported.

As the security type is set at hardware design on a per-master and per-slave basis, it is possible to set some slave access security as configurable from one or some particular masters, and to set the access as Always Secure from all the other masters.

As the security type is set by design at the slave region level, different security region types can be mixed inside a single slave.

Likewise, the mapping base address and the accessible address range of each slave or slave region may have been hardware-restricted on a per-master basis from no access to full slave address space.

#### 18.12.1.2 Examples

The following table shows an example of Security Type settings.

**Table 18-8. Security Type Setting Example**

| Slave | Master0 | Master1 | Master2 |
|---|---|---|---|
| Slave0<br>Internal Memory | Never Secure | Internal Securable<br>1 region | Internal Securable<br>1 region |
| Slave1<br>EBI | External Securable<br>2 regions | Always Secure | External Securable<br>2 regions |

This example is constructed with the following characteristics:

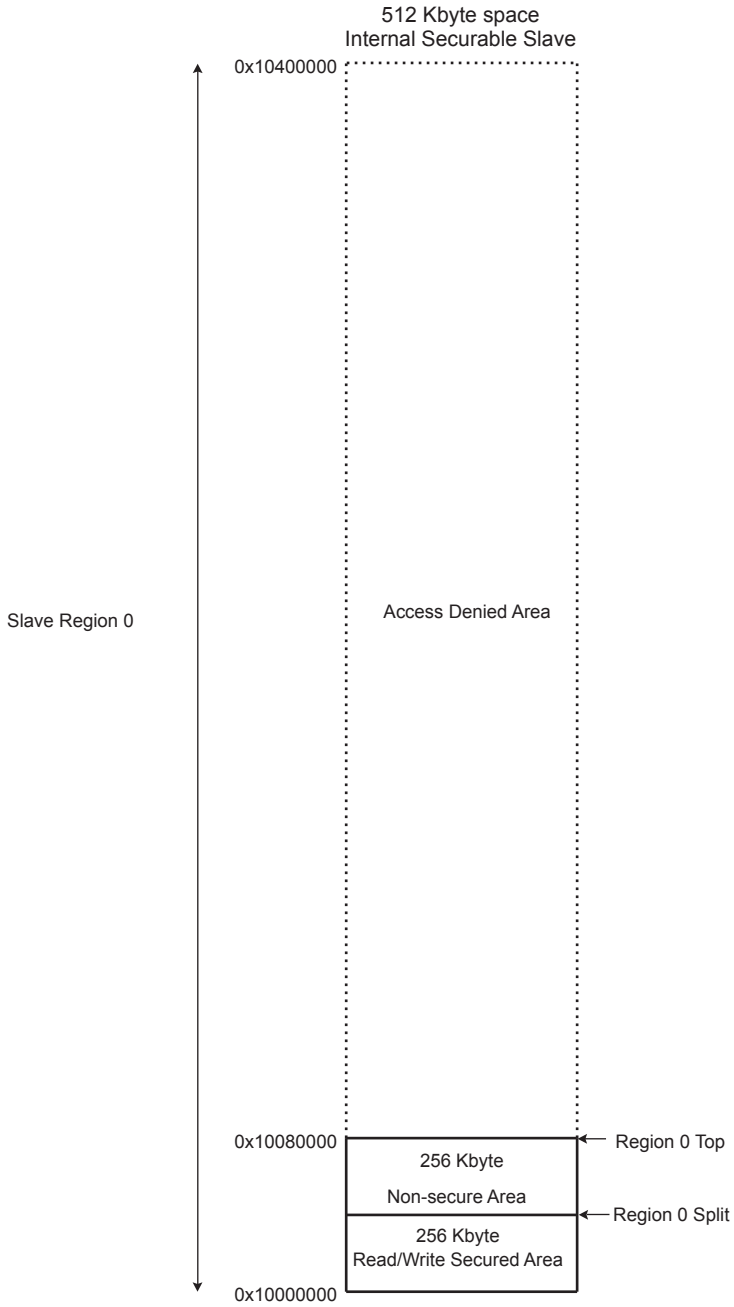
- Slave0 is an Internal Memory containing one region:
  - The Access from Master0 to Slave0 is Never Secure
  - The access from Master1 and Master2 to Slave0 is Internal Securable with one region and with the same software configuration (Choice of SASPLIT0 and the security configuration bits LANSECH, RDNSECH, WRNSECH).
- Slave1 is an EBI containing two regions:
  - The Access from Master1 to Slave1 is Always Secure
  - The access from Master0 and Master2 to Slave1 is External Securable with two regions and with the same software configuration (Choice of SRTOP0, SRTOP1, SASPLIT0, SASPLIT1 and the security configuration bits LANSECH, RDNSECH, WRNSECH).

The figure below shows an Internal Securable slave example. This example is constructed with the following hypothesis:

- The slave is an Internal Memory containing one region. The Slave region max size is 4 Mbytes.
- The slave region 0 base address equals 0x10000000. Its top size is 512 Kbytes (hardware configuration).
- The slave software configuration is:
  - SASPLIT0 is set to 256 Kbytes
  - LANSECH0 is set to 0, the low area of region 0 is the securable one

– RDNSECH0 is set to 0, region 0 Securable area is secured for reads
– WRNSECH0 is set to 0, region 0 Securable area is secured for writes

**Figure 18-2. Partitioning Example of an Internal Securable Slave Featuring 1 Security Region of 512 Kbytes Split into 1 or 2 Security Areas of 4 Kbytes to 512 Kbytes**



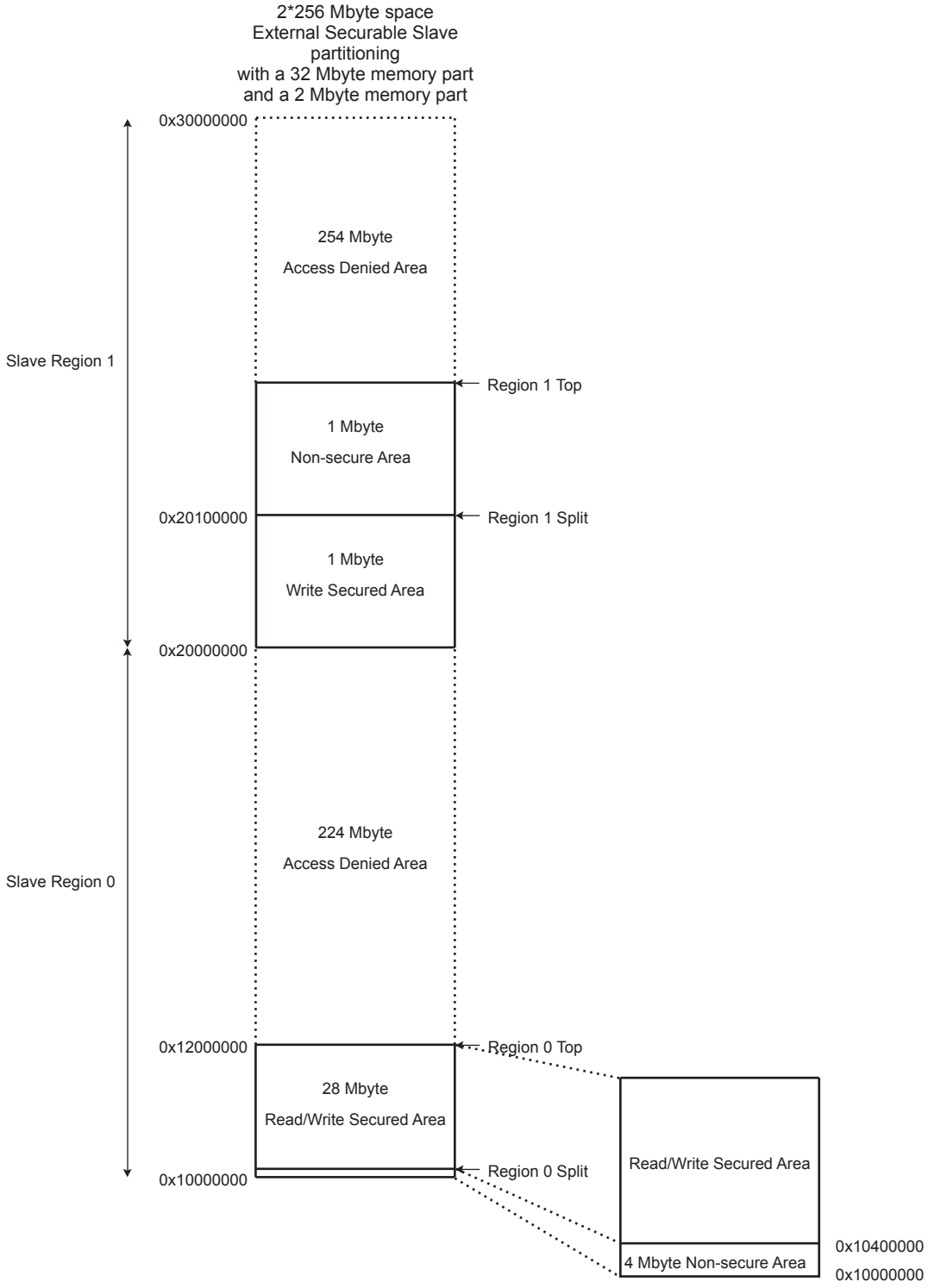

**Note:** The slave security areas split inside the security region are configured by writing into the Security Areas Split Slave Registers.

The figure below shows an External Securable slave example. This example is constructed with the following hypothesis:

- The slave is an interface with the external bus (EBI) containing two regions. The slave size is 2 × 256 Mbytes. Each slave region max size is 256 Mbytes.

- The slave region 0 base address equals 0x10000000. It is connected to a 32 Mbyte memory, for example an external DDR. The slave region 0 top size must be set to 32 Mbytes.
- The slave region 1 base address equals 0x20000000. It is connected to a 2 Mbyte memory, for example an external NAND Flash. The slave region 1 top size must be set to 2 Mbytes.
- The slave software configuration is:
  - SRTOP0 is set to 32 Mbytes
  - SRTOP1 is set to 2 Mbytes
  - SASPLIT0 is set to 4 Mbytes
  - SASPLIT1 is set to 1 Mbyte
  - LANSECH0 is set to 1, the low area of region 0 is the non-securable one
  - RDNSECH0 is set to 0, region 0 Securable area is secured for reads
  - WRNSECH0 is set to 0, region 0 Securable area is secured for writes
  - LANSECH1 is set to 0, the low area of region 1 is the Securable one
  - RDNSECH1 is set to 1, region 1 Securable area is non-secured for reads
  - WRNSECH1 is set to 0, region 1 Securable area is secured for writes

**Figure 18-3. Partitioning Example of an External Securable Slave Featuring 2 Security Regions of 4 Kbytes to 128 Mbytes each and up to 4 Security Areas of 4 Kbytes to 128 Mbytes**



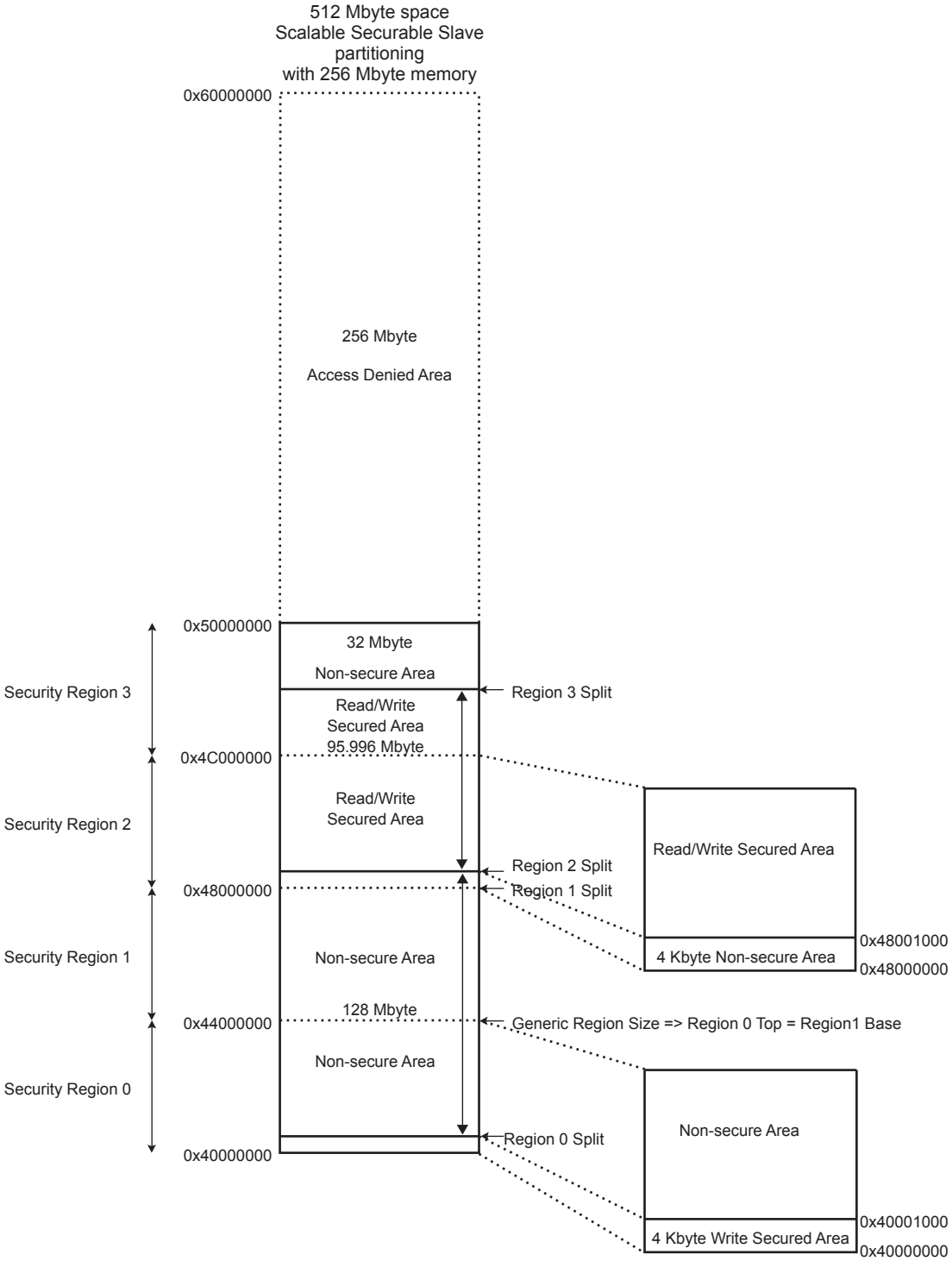

**Note:** The slave region sizes are configured by writing into the Security Region Top Slave Registers.

The slave security area split inside each region is configured by writing into the Security Areas Split Slave Registers.

The figure below shows a Scalable Securable slave example. This example is constructed with the following hypothesis:

- The slave is an external memory with dedicated slave containing four regions, for example an external DDR.
- The slave size is 512 Mbytes.
- The slave base address equals 0x40000000. It is connected to a 256-Mbyte external memory.
- As the connected memory size is 256 Mbytes and there are four regions, the size of each region is 64 Mbytes. This gives the value of the slave region max size and top size. The slave region 0 top size must be configured to 64 Mbytes.
- The slave software configuration is:
  - SRTOP0 is set to 64 Mbytes
  - SASPLIT0 is set to 4 Kbytes
  - SASPLIT1 is set to 64 Mbytes, so its low area occupies the whole region 1
  - SASPLIT2 is set to 4 Kbytes
  - SASPLIT3 is set to 32 Mbytes
  - LANSECH0 is set to 0, the low area of region 0 is the Securable one
  - RDNSECH0 is set to 1, region 0 Securable area is non-secured for reads
  - WRNSECH0 is set to 0, region 0 Securable area is secured for writes
  - LANSECH1 is set to 1, the low area of region 1 is the non-securable one
  - RDNSECH1 is ‘don’t care’ since the low area occupies the whole region 1
  - WRNSECH1 is ‘don’t care’ since the low area occupies the whole region 1
  - LANSECH2 is set to 1, the low area of region 2 is the non-securable one
  - RDNSECH2 is set to 0, region 2 Securable area is secured for reads
  - WRNSECH2 is set to 0, region 2 Securable area is secured for writes
  - LANSECH3 is set to 0, the low area of region 3 is the Securable one
  - RDNSECH3 is set to 0, region 3 Securable area is secured for reads
  - WRNSECH3 is set to 0, region 3 Securable area is secured for writes

**Figure 18-4. Partitioning Example of a Scalable Securable Slave Featuring 4 Equally-sized Security Regions of 1 Mbytes to 128 Mbytes each and up to 8 Security Areas of 4 Kbytes to 128 Mbytes**




**Note:** The slaves' generic security regions sizes are configured by writing into field SRTOP0 of the Security Region Top Slave Registers and the custom slave security areas splits inside each region is configured by writing into the Security Areas Split Slave Registers.

### 18.12.2 Security Types of SDMMC System Bus Slaves

The SDMMC user interface is connected as a system bus slave, and must be configured as Internal Securable to Peripheral (ISP).

Each region in the “Internal Securable to Peripheral” slave type must be programmed with the following characteristics:

- The region must be programmed to be entirely secure or entirely non-secure. This is done with:
  - The split offset must be equal to the maximum size of 128 Mbytes so that the whole peripheral user interface is in the low area below the split. Code sample: MATRIX_SASSRx.SASPLITy = 0xF
  - The bits WRNSECH and RDNSECH must be set respectively to 0=”write secured” and 0=”read secured”. Code sample: MATRIX_SSRx.WRNSECHy = 0; MATRIX_SSRx.RDNSECHy = 0;
  - To set the peripheral to “secure”: the bit LANSECHy must be set to 0 (low area according to RDNSECH0 and WRNSECH0, hence secure).
  - To set the peripheral to “non-secure”: the bit LANSECHy must be set to 1 (low area is non-secure).
    **Note:** The MATRIX_SRTSRx register is not applicable for the “Internal Securable to Peripheral” type.
- The Security Peripheral Select registers (MATRIX_SPSELRx) must be set to the same security attributes for the corresponding Peripheral identifiers: MATRIX_SPSELRx.NSECPy.

### 18.12.3 Security Types of System Bus Masters

Masters send requests to the MATRIX with a security attribute that depends on the master security type, which is identical to the security type of the slave user interface.

For DMA, the TrustZone security attribute can be selected for each channel. Refer to the XDMAC user interface description.

### 18.12.4 Security of Peripheral Bus Slaves

The security type of an APB slave is set at hardware design among the following:

- Always Secure (AS)
- Never Secure (NS)
- Programmable Secure (PS)

To configure the security mode required for accessing a peripheral bus slave connected to the system-to-peripheral bus bridge (HBRIDGE), the MATRIX features three 32-bit Security Peripheral Select x Registers. Some of these bits may have been set to a secure or a non-secure value by design, whereas others are programmed by software (see “Security Peripheral Select x Registers”).

Peripheral security state, “secure” or “non-secure” is an AND operation between H32MX MATRIX_SPSELRx and H64MX MATRIX_SPSELRx for the bit corresponding to the peripheral.

As a general rule:

- The peripheral security state is applied to the corresponding peripheral interrupt line. Exceptions may occur on some peripherals (PIO Controller, etc.). In such case, refer to the peripheral description.
- The peripheral security state is applied to the peripheral master part, if any. Exceptions may occur on some peripherals. In such case, refer to the peripheral description. See “Security Types of AHB Master Peripherals”.

MATRIX_SPSELRx bits in the H32MX or H64MX user interface are respectively read/write or read-only to ‘1’ depending on whether the peripheral is connected or not, on the matrix.

All bit values in the following table except those marked ‘UD’ (User Defined) are read-only and cannot be changed. Values marked ‘UD’ can be changed. Refer to the following examples.

- Example for GMAC, Peripheral ID 5, which is connected to the H32MX Matrix
  - H64MX MATRIX_SPSELR1[5] = 1 (read-only); no influence on the security configuration
  - H32MX MATRIX_SPSELR1[5] can be written by user to program the security.
- Example for LCDC, Peripheral ID 45, which is connected to the H64MX Matrix
  - H64MX MATRIX_SPSELR2[13] can be written by user to program the security.
  - H32MX MATRIX_SPSELR2[13] = 1 (read-only); no influence on the security configuration
- Example for AIC, Peripheral ID 49, which is connected to the H32MX Matrix
  - H64MX MATRIX_SPSELR2[17] = 1 (read-only); sets the peripheral as Non-secure by hardware, also called “Never Secure”
  - H32MX MATRIX_SPSELR2[17] = 1 (read-only); no influence on the security configuration
- Example for SAIC, Peripheral ID 0, which is connected to the H32MX Matrix
  - H64MX MATRIX_SPSELR1[0] = 1 (read-only); no influence on the security configuration
  - H32MX MATRIX_SPSELR1[0] = 0 (read-only); sets the peripheral as Secure by hardware, also called “Always Secure”

**Table 18-9. Peripheral Identifiers**

| Instance ID | Instance Name | Internal Interrupt | PMC Clock Control | Instance Description | Clock Type | Security(1) | In Matrix | MATRIX_SPSELRx Bit | Bit Value in H32MX | Bit Value in H64MX |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | SAIC | FIQ | – | FIQ Interrupt ID | SYS_CLK_LS | AS | – | MATRIX_SPSELR1[0] | 0 | 1 |
| 1 | PMC | X | – | Power Management Controller | SYS_CLK_LS | PS | H32MX | MATRIX_SPSELR1[1] | UD | 1 |
| 2 | ARM | PMU | X | Performance Monitor Unit (PMU) | PROC_CLK | PS | H64MX | MATRIX_SPSELR1[2] | 1 | UD |
| 3 | PIT | X | – | Periodic Interval Timer Interrupt | SYS_CLK_LS | PS(3) | H32MX | – | – | – |
| 4 | WDT | X | – | Watchdog Timer Interrupt | SYS_CLK_LS | PS(3) | H32MX | – | – | – |
| 5 | GMAC | X | X | Ethernet MAC | HCLOCK_LS PCLOCK_LS | PS | H32MX | MATRIX_SPSELR1[5] | UD | 1 |
| 6 | XDMAC0 | X | X | DMA Controller 0 | HCLOCK_HS | PS | H64MX | MATRIX_SPSELR1[6] | 1 | UD |
| 7 | XDMAC1 | X | X | DMA Controller 1 | HCLOCK_HS | PS | H64MX | MATRIX_SPSELR1[7] | 1 | UD |
| 8 | ICM | X | X | Integrity Check Monitor | HCLOCK_LS | PS | H32MX | MATRIX_SPSELR1[8] | UD | 1 |
| 9 | AES | X | X | Advanced Encryption Standard | PCLK_HS | PS | H64MX | MATRIX_SPSELR1[9] | 1 | UD |
| 10 | AESB | X | X | AES Bridge | HCLOCK_HS | PS | H64MX | MATRIX_SPSELR1[10] | 1 | UD |
| 11 | TDES | X | X | Triple Data Encryption Standard | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR1[11] | UD | 1 |
| 12 | SHA | X | X | SHA Signature | PCLK_HS | PS | H64MX | MATRIX_SPSELR1[12] | 1 | UD |
| 13 | MPDDRC | X | X | MPDDR Controller | HCLOCK_HS | PS | H64MX | MATRIX_SPSELR1[13] | 1 | UD |
| 14 | H32MX | X | X | 32-bit Matrix | SYS_CLK_LS | AS | – | MATRIX_SPSELR1[14] | 0 | 1 |
| 15 | H64MX | X | X | 64-bit Matrix | SYS_CLOCK | AS | – | MATRIX_SPSELR1[15] | 1 | 0 |
| 16 | SECUMOD | X | X | Security Module | SLOW_CLOCK | AS | H32MX | MATRIX_SPSELR1[16] | 0 | 1 |
| 17 | HSMC | X | X | Multibit ECC Interrupt | HCLOCK_LS | PS | H32MX | MATRIX_SPSELR1[17] | UD | 1 |
| 18 | PIOA | X | X | Parallel I/O Controller | PCLOCK_LS | AS | H32MX | MATRIX_SPSELR1[18] | 0 | 1 |
| 19 | FLEXCOM0 | X | X | FLEXCOM 0 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR1[19] | UD | 1 |
| 20 | FLEXCOM1 | X | X | FLEXCOM 1 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR1[20] | UD | 1 |
| 21 | FLEXCOM2 | X | X | FLEXCOM 2 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR1[21] | UD | 1 |
| 22 | FLEXCOM3 | X | X | FLEXCOM 3 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR1[22] | UD | 1 |

© 2018 Microchip Technology Inc.

.....continued

| Instance ID | Instance Name | Internal Interrupt | PMC Clock Control | Instance Description | Clock Type | Security(1) | In Matrix | MATRIX_SPSELRx Bit | Bit Value in H32MX | Bit Value in H64MX |
|---|---|---|---|---|---|---|---|---|---|---|
| 23 | FLEXCOM4 | X | X | FLEXCOM 4 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR1[23] | UD | 1 |
| 24 | UART0 | X | X | Universal Asynchronous Receiver Transmitter 0 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR1[24] | UD | 1 |
| 25 | UART1 | X | X | Universal Asynchronous Receiver Transmitter 1 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR1[25] | UD | 1 |
| 26 | UART2 | X | X | Universal Asynchronous Receiver Transmitter 2 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR1[26] | UD | 1 |
| 27 | UART3 | X | X | Universal Asynchronous Receiver Transmitter 3 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR1[27] | UD | 1 |
| 28 | UART4 | X | X | Universal Asynchronous Receiver Transmitter 4 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR1[28] | UD | 1 |
| 29 | TWIHS0 | X | X | Two-Wire Interface 0 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR1[29] | UD | 1 |
| 30 | TWIHS1 | X | X | Two-Wire Interface 1 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR1[30] | UD | 1 |
| 31 | SDMMC0 | X | X | Secure Digital MultiMedia Card Controller 0 | HCLOCK_HS | PS | H64MX | MATRIX_SPSELR1[31] | 1 | UD |
| 32 | SDMMC1 | X | X | Secure Digital MultiMedia Card Controller 1 | HCLOCK_HS | PS | H64MX | MATRIX_SPSELR2[0] | 1 | UD |
| 33 | SPI0 | X | X | Serial Peripheral Interface 0 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[1] | UD | 1 |
| 34 | SPI1 | X | X | Serial Peripheral Interface 1 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[2] | UD | 1 |
| 35 | TC0 | X | X | Timer Counter 0 (ch. 0, 1, 2) | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[3] | UD | 1 |
| 36 | TC1 | X | X | Timer Counter 1 (ch. 3, 4, 5) | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[4] | UD | 1 |
| 37 | – | – | – | – | – | – | – | – | – | – |
| 38 | PWM | X | X | Pulse Width Modulation Controller 0 (ch. 0, 1, 2, 3) | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[6] | UD | 1 |
| 39 | – | – | – | – | – | – | – | – | – | – |
| 40 | ADC | X | X | Touchscreen ADC Controller | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[8] | UD | 1 |
| 41 | UHPHS | X | X | USB Host High-Speed | HCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[9] | UD | 1 |

© 2018 Microchip Technology Inc.

.....continued

| Instance ID | Instance Name | Internal Interrupt | PMC Clock Control | Instance Description | Clock Type | Security(1) | In Matrix | MATRIX_SPSELRx Bit | Bit Value in H32MX | Bit Value in H64MX |
|---|---|---|---|---|---|---|---|---|---|---|
| 42 | UDPHS | X | X | USB Device High-Speed | HCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[10] | UD | 1 |
| 43 | SSC0 | X | X | Synchronous Serial Controller 0 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[11] | UD | 1 |
| 44 | SSC1 | X | X | Synchronous Serial Controller 1 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[12] | UD | 1 |
| 45 | LCDC | X | X | LCD Controller | HCLOCK_HS | PS | H64MX | MATRIX_SPSELR2[13] | 1 | UD |
| 46 | ISC | X | X | Image Sensor Controller | HCLOCK_HS | PS | H64MX | MATRIX_SPSELR2[14] | 1 | UD |
| 47 | TRNG | X | X | True Random Number Generator | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[15] | UD | 1 |
| 48 | PDMIC | X | X | Pulse Density Modulation Interface Controller | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[16] | UD | 1 |
| 49 | AIC | IRQ | – | IRQ Interrupt ID | SYS_CLK_LS | NS | H32MX | MATRIX_SPSELR2[17] | 1 | 1 |
| 50 | SFC | X | X | Secure Fuse Controller | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[18] | UD | 1 |
| 51 | SECURAM | X | X | Secure RAM | PCLOCK_LS | AS | H32MX | MATRIX_SPSELR2[19] | 0 | 1 |
| 52 | QSPI0 | X | X | Quad SPI Interface 0 | HCLOCK_HS | PS | H64MX | MATRIX_SPSELR2[20] | 1 | UD |
| 53 | QSPI1 | X | X | Quad SPI Interface 1 | HCLOCK_HS | PS | H64MX | MATRIX_SPSELR2[21] | 1 | UD |
| 54 | I2SC0 | X | X | Inter-IC Sound Controller 0 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[22] | UD | 1 |
| 55 | I2SC1 | X | X | Inter-IC Sound Controller 1 | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[23] | UD | 1 |
| 56 | MCAN0 | INT0 | X | MCAN 0 Interrupt0 | HCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[24] | UD | 1 |
| 57 | MCAN1 | INT0 | X | MCAN 1 Interrupt0 | HCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[25] | UD | 1 |
| 58 | PTC | X | X | Peripheral Touch Controller | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[26] | UD | 1 |
| 59 | CLASSD | X | X | Audio Class D Amplifier | PCLOCK_LS | PS | H32MX | MATRIX_SPSELR2[27] | UD | 1 |
| 60 | SFR | – | – | Special Function Register(2) | SYS_CLK_LS | PS | H32MX | MATRIX_SPSELR2[28] | UD | 1 |
| 61 | SAIC | – | – | Secure Advanced Interrupt Controller(2) | SYS_CLK_LS | AS | H32MX | MATRIX_SPSELR2[29] | 0 | 1 |
| 62 | AIC | – | – | Advanced Interrupt Controller(2) | SYS_CLK_LS | NS | H32MX | MATRIX_SPSELR2[30] | 1 | 1 |
| 63 | L2CC | X | – | L2 Cache Controller | – | PS | H64MX | MATRIX_SPSELR2[31] | 1 | UD |

© 2018 Microchip Technology Inc.

.....continued

| Instance ID | Instance Name | Internal Interrupt | PMC Clock Control | Instance Description | Clock Type | Security(1) | In Matrix | MATRIX_SPSELRx Bit | Bit Value in H32MX | Bit Value in H64MX |
|---|---|---|---|---|---|---|---|---|---|---|
| 64 | MCAN0 | INT1 | – | MCAN 0 Interrupt1 | – | PS | H32MX | MATRIX_SPSELR3[0] | UD | 1 |
| 65 | MCAN1 | INT1 | – | MCAN 1 Interrupt1 | – | PS | H32MX | MATRIX_SPSELR3[1] | UD | 1 |
| 66 | GMAC | Q1 | – | GMAC Queue 1 Interrupt | – | PS | H32MX | MATRIX_SPSELR3[2] | UD | 1 |
| 67 | GMAC | Q2 | – | GMAC Queue 2 Interrupt | – | PS | H32MX | MATRIX_SPSELR3[3] | UD | 1 |
| 68 | PIOB | X | – | – | – | AS | H32MX | MATRIX_SPSELR3[4] | 0 | 1 |
| 69 | PIOC | X | – | – | – | AS | H32MX | MATRIX_SPSELR3[5] | 0 | 1 |
| 70 | PIOD | X | – | – | – | AS | H32MX | MATRIX_SPSELR3[6] | 0 | 1 |
| 71 | SDMMC0 | TIMER | – | – | – | PS | H32MX | MATRIX_SPSELR3[7] | UD | 1 |
| 72 | SDMMC1 | TIMER | – | – | – | PS | H32MX | MATRIX_SPSELR3[8] | UD | 1 |
| 73 | RSTC | X | – | Reset Controller | SYS_CLK_LS | PS(3) | H32MX | – | – | – |
| 74 | SYSC, RTC | X | – | System Controller Interrupt | SYS_CLK_LS | PS(3) | H32MX | MATRIX_SPSELR3[10] | UD | 1 |
| 75 | ACC | X | – | Analog Comparator | SYS_CLK_LS | PS | H32MX | MATRIX_SPSELR3[11] | UD | 1 |
| 76 | RXLP | X | – | UART Low-Power | SYS_CLK_LS | PS | H32MX | MATRIX_SPSELR3[12] | UD | 1 |
| 77 | SFRBU | – | – | Special Function Register Backup(2) | – | PS | H32MX | MATRIX_SPSELR3[13] | UD | 1 |
| 78 | CHIPID | – | – | Chip ID | – | PS | H32MX | MATRIX_SPSELR3[14] | UD | 1 |

© 2018 Microchip Technology Inc.

**Note:**

1. AS = Always Secure; PS = Programmable Secure; NS = Never Secure.
2. For security purposes, there is no matching clock but a peripheral ID only.
3. The PIT, RSTC and WDT are controlled by the RTC. They are in Secure mode if the RTC is in Secure mode; they are in Non-secure mode if the RTC is in Non-secure mode.

The system-to-peripheral bus bridge compares the incoming master request security bit with the required security mode for the selected peripheral, and accepts or denies access. In the last case, its bus error response is internally flagged in the Master Error Status Register; the offending address is registered in the Master Error Address Registers so that the slave and the targeted protected region are also known.

## 18.13 Register Summary

The user interface below is constructed with the maximum numbers of masters, slaves and regions by slave that are possible on the two product matrixes. The exact number of these elements must be used to deduce the exact register description of the Matrix user interface.

The exact numbers of these elements can be found in the following sections:

- "64-bit Matrix (H64MX)"
- "32-bit Matrix (H32MX)"

| Offset | Name | Bit Pos. | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | MATRIX_MCFG0 | 7:0 | | | | | | ULBT[2:0] | | | |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| ... | | | | | | | | | | | |
| 0x2C | MATRIX_MCFG11 | 7:0 | | | | | | ULBT[2:0] | | | |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x30 ... 0x3F | Reserved | | | | | | | | | | |
| 0x40 | MATRIX_SCFG0 | 7:0 | SLOT_CYCLE[7:0] | | | | | | | | |
| | | 15:8 | | | | | | | | SLOT_CYCLE[8:8] | |
| | | 23:16 | | | FIXED_DEFMSTR[3:0] | | | | DEFMSTR_TYPE[1:0] | | |
| | | 31:24 | | | | | | | | | |
| ... | | | | | | | | | | | |
| 0x78 | MATRIX_SCFG14 | 7:0 | SLOT_CYCLE[7:0] | | | | | | | | |
| | | 15:8 | | | | | | | | SLOT_CYCLE[8:8] | |
| | | 23:16 | | | FIXED_DEFMSTR[3:0] | | | | DEFMSTR_TYPE[1:0] | | |
| | | 31:24 | | | | | | | | | |
| 0x7C ... 0x7F | Reserved | | | | | | | | | | |
| 0x80 | MATRIX_PRAS0 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | | |
| 0x84 | MATRIX_PRBS0 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x88 | MATRIX_PRAS1 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0x8C | MATRIX_PRBS1 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x90 | MATRIX_PRAS2 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0x94 | MATRIX_PRBS2 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x98 | MATRIX_PRAS3 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0x9C | MATRIX_PRBS3 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xA0 | MATRIX_PRAS4 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0xA4 | MATRIX_PRBS4 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xA8 | MATRIX_PRAS5 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0xAC | MATRIX_PRBS5 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xB0 | MATRIX_PRAS6 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xB4 | MATRIX_PRBS6 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xB8 | MATRIX_PRAS7 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0xBC | MATRIX_PRBS7 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xC0 | MATRIX_PRAS8 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0xC4 | MATRIX_PRBS8 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xC8 | MATRIX_PRAS9 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0xCC | MATRIX_PRBS9 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xD0 | MATRIX_PRAS10 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0xD4 | MATRIX_PRBS10 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xD8 | MATRIX_PRAS11 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0xDC | MATRIX_PRBS11 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xE0 | MATRIX_PRAS12 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0xE4 | MATRIX_PRBS12 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xE8 | MATRIX_PRAS13 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0xEC | MATRIX_PRBS13 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xF0 | MATRIX_PRAS14 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0xF4 | MATRIX_PRBS14 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xF8 ... 0x014F | Reserved | | | | | | | | | |
| 0x0150 | MATRIX_MEIER | 7:0 | MERR7 | MERR6 | MERR5 | MERR4 | MERR3 | MERR2 | MERR1 | MERR0 |
| | | 15:8 | | | | | MERR11 | MERR10 | MERR9 | MERR8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0154 | MATRIX_MEIDR | 7:0 | MERR7 | MERR6 | MERR5 | MERR4 | MERR3 | MERR2 | MERR1 | MERR0 |
| | | 15:8 | | | | | MERR11 | MERR10 | MERR9 | MERR8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0158 | MATRIX_MEIMR | 7:0 | MERR7 | MERR6 | MERR5 | MERR4 | MERR3 | MERR2 | MERR1 | MERR0 |
| | | 15:8 | | | | | MERR11 | MERR10 | MERR9 | MERR8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x015C | MATRIX_MESR | 7:0 | MERR7 | MERR6 | MERR5 | MERR4 | MERR3 | MERR2 | MERR1 | MERR0 |
| | | 15:8 | | | | | MERR11 | MERR10 | MERR9 | MERR8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0160 | MATRIX_MEAR0 | 7:0 | ERRADD[7:0] | | | | | | | |
| | | 15:8 | ERRADD[15:8] | | | | | | | |
| | | 23:16 | ERRADD[23:16] | | | | | | | |
| | | 31:24 | ERRADD[31:24] | | | | | | | |
| ... | | | | | | | | | | |
| 0x018C | MATRIX_MEAR11 | 7:0 | ERRADD[7:0] | | | | | | | |
| | | 15:8 | ERRADD[15:8] | | | | | | | |
| | | 23:16 | ERRADD[23:16] | | | | | | | |
| | | 31:24 | ERRADD[31:24] | | | | | | | |
| 0x0190 ... 0x01E3 | Reserved | | | | | | | | | |
| 0x01E4 | MATRIX_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0x01E8 | MATRIX_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01EC ... 0x01FF | Reserved | | | | | | | | | |
| 0x0200 | MATRIX_SSR0 | 7:0 | LANSECH7 | LANSECH6 | LANSECH5 | LANSECH4 | LANSECH3 | LANSECH2 | LANSECH1 | LANSECH0 |
| | | 15:8 | RDNSECH7 | RDNSECH6 | RDNSECH5 | RDNSECH4 | RDNSECH3 | RDNSECH2 | RDNSECH1 | RDNSECH0 |
| | | 23:16 | WRNSECH7 | WRNSECH6 | WRNSECH5 | WRNSECH4 | WRNSECH3 | WRNSECH2 | WRNSECH1 | WRNSECH0 |
| | | 31:24 | | | | | | | | |
| ... | | | | | | | | | | |
| 0x0238 | MATRIX_SSR14 | 7:0 | LANSECH7 | LANSECH6 | LANSECH5 | LANSECH4 | LANSECH3 | LANSECH2 | LANSECH1 | LANSECH0 |
| | | 15:8 | RDNSECH7 | RDNSECH6 | RDNSECH5 | RDNSECH4 | RDNSECH3 | RDNSECH2 | RDNSECH1 | RDNSECH0 |
| | | 23:16 | WRNSECH7 | WRNSECH6 | WRNSECH5 | WRNSECH4 | WRNSECH3 | WRNSECH2 | WRNSECH1 | WRNSECH0 |
| | | 31:24 | | | | | | | | |
| 0x023C ... 0x023F | Reserved | | | | | | | | | |
| 0x0240 | MATRIX_SASSR0 | 7:0 | SASPLIT1[3:0] | | | | SASPLIT0[3:0] | | | |
| | | 15:8 | SASPLIT3[3:0] | | | | SASPLIT2[3:0] | | | |
| | | 23:16 | SASPLIT5[3:0] | | | | SASPLIT4[3:0] | | | |
| | | 31:24 | SASPLIT7[3:0] | | | | SASPLIT6[3:0] | | | |
| ... | | | | | | | | | | |
| 0x0278 | MATRIX_SASSR14 | 7:0 | SASPLIT1[3:0] | | | | SASPLIT0[3:0] | | | |
| | | 15:8 | SASPLIT3[3:0] | | | | SASPLIT2[3:0] | | | |
| | | 23:16 | SASPLIT5[3:0] | | | | SASPLIT4[3:0] | | | |
| | | 31:24 | SASPLIT7[3:0] | | | | SASPLIT6[3:0] | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x027C ... 0x0283 | Reserved | | | | | | | | | |
| 0x0284 | MATRIX_SRTSR1 | 7:0 | SRTOP1[3:0] | | | | SRTOP0[3:0] | | | |
| | | 15:8 | SRTOP3[3:0] | | | | SRTOP2[3:0] | | | |
| | | 23:16 | SRTOP5[3:0] | | | | SRTOP4[3:0] | | | |
| | | 31:24 | SRTOP7[3:0] | | | | SRTOP6[3:0] | | | |
| ... | | | | | | | | | | |
| 0x02BC | MATRIX_SRTSR15 | 7:0 | SRTOP1[3:0] | | | | SRTOP0[3:0] | | | |
| | | 15:8 | SRTOP3[3:0] | | | | SRTOP2[3:0] | | | |
| | | 23:16 | SRTOP5[3:0] | | | | SRTOP4[3:0] | | | |
| | | 31:24 | SRTOP7[3:0] | | | | SRTOP6[3:0] | | | |
| 0x02C0 | MATRIX_SPSELR1 | 7:0 | NSECP7 | NSECP6 | NSECP5 | NSECP4 | NSECP3 | NSECP2 | NSECP1 | NSECP0 |
| | | 15:8 | NSECP15 | NSECP14 | NSECP13 | NSECP12 | NSECP11 | NSECP10 | NSECP9 | NSECP8 |
| | | 23:16 | NSECP23 | NSECP22 | NSECP21 | NSECP20 | NSECP19 | NSECP18 | NSECP17 | NSECP16 |
| | | 31:24 | NSECP31 | NSECP30 | NSECP29 | NSECP28 | NSECP27 | NSECP26 | NSECP25 | NSECP24 |
| 0x02C4 | MATRIX_SPSELR2 | 7:0 | NSECP7 | NSECP6 | NSECP5 | NSECP4 | NSECP3 | NSECP2 | NSECP1 | NSECP0 |
| | | 15:8 | NSECP15 | NSECP14 | NSECP13 | NSECP12 | NSECP11 | NSECP10 | NSECP9 | NSECP8 |
| | | 23:16 | NSECP23 | NSECP22 | NSECP21 | NSECP20 | NSECP19 | NSECP18 | NSECP17 | NSECP16 |
| | | 31:24 | NSECP31 | NSECP30 | NSECP29 | NSECP28 | NSECP27 | NSECP26 | NSECP25 | NSECP24 |
| 0x02C8 | MATRIX_SPSELR3 | 7:0 | NSECP7 | NSECP6 | NSECP5 | NSECP4 | NSECP3 | NSECP2 | NSECP1 | NSECP0 |
| | | 15:8 | NSECP15 | NSECP14 | NSECP13 | NSECP12 | NSECP11 | NSECP10 | NSECP9 | NSECP8 |
| | | 23:16 | NSECP23 | NSECP22 | NSECP21 | NSECP20 | NSECP19 | NSECP18 | NSECP17 | NSECP16 |
| | | 31:24 | NSECP31 | NSECP30 | NSECP29 | NSECP28 | NSECP27 | NSECP26 | NSECP25 | NSECP24 |

### 18.13.1 Bus Matrix Master Configuration Registers

**Name:** MATRIX_MCFGx
**Offset:** 0x00 + x*0x04 [x=0..11]
**Reset:** 0x00000004
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | ULBT[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 1 | 0 | 0 |

**Bits 2:0 – ULBT[2:0]** Undefined Length Burst Type

| Value | Name | Description |
|---|---|---|
| 0 | UNLIMITED | Unlimited Length Burst—No predicted end of burst is generated, therefore INCR bursts coming from this master can only be broken if the Slave Slot Cycle Limit is reached. If the Slot Cycle Limit is not reached, the burst is normally completed by the master, at the latest, on the next system bus 1 Kbyte address boundary, allowing up to 256-beat word bursts or 128-beat double-word bursts.<br>This value should not be used in the very particular case of a master capable of performing back-to-back undefined length bursts on a single slave, since this could indefinitely freeze the slave arbitration and thus prevent another master from accessing this slave. |
| 1 | SINGLE | Single Access—The undefined length burst is treated as a succession of single accesses, allowing rearbitration at each beat of the INCR burst or bursts sequence. |
| 2 | 4_BEAT | 4-beat Burst—The undefined length burst or bursts sequence is split into 4-beat bursts or less, allowing rearbitration every 4 beats. |
| 3 | 8_BEAT | 8-beat Burst—The undefined length burst or bursts sequence is split into 8-beat bursts or less, allowing rearbitration every 8 beats. |

| Value | Name | Description |
|---|---|---|
| 4 | 16_BEAT | 16-beat Burst—The undefined length burst or bursts sequence is split into 16-beat bursts or less, allowing rearbitration every 16 beats. |
| 5 | 32_BEAT | 32-beat Burst—The undefined length burst or bursts sequence is split into 32-beat bursts or less, allowing rearbitration every 32 beats. |
| 6 | 64_BEAT | 64-beat Burst—The undefined length burst or bursts sequence is split into 64-beat bursts or less, allowing rearbitration every 64 beats. |
| 7 | 128_BEAT | 128-beat Burst—The undefined length burst or bursts sequence is split into 128-beat bursts or less, allowing rearbitration every 128 beats.<br>Unless duly needed, the ULBT should be left at its default 0 value for power saving. |

### 18.13.2 Bus Matrix Slave Configuration Registers

**Name:** MATRIX_SCFGx
**Offset:** 0x40 + x*0x04 [x=0..14]
**Reset:** 0x00000004
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | FIXED_DEFMSTR[3:0] | | | | DEFMSTR_TYPE[1:0] | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SLOT_CYCLE[8:8] |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SLOT_CYCLE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

**Bits 21:18 – FIXED_DEFMSTR[3:0]** Fixed Default Master
This is the number of the Default Master for this slave. Only used if DEFMSTR_TYPE value = 2. Specifying the number of a master which is not connected to the selected slave is equivalent to clearing DEFMSTR_TYPE.

**Bits 17:16 – DEFMSTR_TYPE[1:0]** Default Master Type

| Value | Name | Description |
|---|---|---|
| 0 | NONE | No Default Master—At the end of the current slave access, if no other master request is pending, the slave is disconnected from all masters.<br>This results in a one clock cycle latency for the first access of a burst transfer or for a single access. |
| 1 | LAST | Last Default Master—At the end of the current slave access, if no other master request is pending, the slave stays connected to the last master having accessed it.<br>This results in not having one clock cycle latency when the last master tries to access the slave again. |

| Value | Name | Description |
|---|---|---|
| 2 | FIXED | Fixed Default Master—At the end of the current slave access, if no other master request is pending, the slave connects to the fixed master the number that has been written in the FIXED_DEFMSTR field.<br>This results in not having one clock cycle latency when the fixed master tries to access the slave again. |

**Bits 8:0 – SLOT_CYCLE[8:0]** Maximum Bus Grant Duration for Masters
When SLOT_CYCLE system bus clock cycles have elapsed since the last arbitration, a new arbitration takes place to let another master access this slave. If another master is requesting the slave bus, then the current master burst is broken.

If SLOT_CYCLE = 0, the Slot Cycle Limit feature is disabled and bursts always complete unless broken according to the ULBT.

This limit has been placed in order to enforce arbitration so as to meet potential latency constraints of masters waiting for slave access.

This limit must not be too small. Unreasonably small values break every burst and the MATRIX arbitrates without performing any data transfer. The default maximum value is usually an optimal conservative choice.

In most cases, this feature is not needed and should be disabled for power saving.

See “Slot Cycle Limit Arbitration” for details.

### 18.13.3 Bus Matrix Priority Registers A For Slaves

**Name:** MATRIX_PRASx
**Offset:** 0x80 + x*0x08 [x=0..14]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | M7PR[1:0] | | | | M6PR[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | M5PR[1:0] | | | | M4PR[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | M3PR[1:0] | | | | M2PR[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | M1PR[1:0] | | | | M0PR[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MPR** Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.

All the masters programmed with the same MxPR value for the slave make up a priority pool.

Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.

Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).

See “Arbitration Priority Scheme” for details.

### 18.13.4 Bus Matrix Priority Registers B For Slaves

**Name:** MATRIX_PRBSx
**Offset:** 0x84 + x*0x08 [x=0..14]
**Reset:** 0
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | M3PR[1:0] | | | | M2PR[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | M1PR[1:0] | | | | M0PR[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bits 0:1, 4:5, 8:9, 12:13 – MPR** Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.

All the masters programmed with the same MxPR value for the slave make up a priority pool.

Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.

Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).

See “Arbitration Priority Scheme” for details.

### 18.13.5 Master Error Interrupt Enable Register

**Name:** MATRIX_MEIER
**Offset:** 0x0150
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MERR11 | MERR10 | MERR9 | MERR8 |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MERR7 | MERR6 | MERR5 | MERR4 | MERR3 | MERR2 | MERR1 | MERR0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – MERRx** Master x Access Error

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables Master x Access Error interrupt source. |

### 18.13.6 Master Error Interrupt Disable Register

**Name:** MATRIX_MEIDR
**Offset:** 0x0154
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MERR11 | MERR10 | MERR9 | MERR8 |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MERR7 | MERR6 | MERR5 | MERR4 | MERR3 | MERR2 | MERR1 | MERR0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – MERRx** Master x Access Error

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables Master x Access Error interrupt source. |

### 18.13.7 Master Error Interrupt Mask Register

**Name:** MATRIX_MEIMR
**Offset:** 0x0158
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MERR11 | MERR10 | MERR9 | MERR8 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MERR7 | MERR6 | MERR5 | MERR4 | MERR3 | MERR2 | MERR1 | MERR0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – MERRx** Master x Access Error

| Value | Description |
|---|---|
| 0 | Master x Access Error does not trigger any interrupt. |
| 1 | Master x Access Error triggers the MATRIX interrupt line. |

### 18.13.8 Master Error Status Register

**Name:** MATRIX_MESR
**Offset:** 0x015C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MERR11 | MERR10 | MERR9 | MERR8 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MERR7 | MERR6 | MERR5 | MERR4 | MERR3 | MERR2 | MERR1 | MERR0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – MERRx** Master x Access Error

| Value | Description |
|---|---|
| 0 | No Master Access Error has occurred since the last read of the MATRIX_MESR. |
| 1 | At least one Master Access Error has occurred since the last read of the MATRIX_MESR. |

### 18.13.9 Master Error Address Registers

**Name:** MATRIX_MEARx
**Offset:** 0x0160 + x*0x04 [x=0..11]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ERRADD[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ERRADD[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ERRADD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERRADD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ERRADD[31:0]** Master Error Address
Master Last Access Error Address

### 18.13.10 Write Protection Mode Register

**Name:** MATRIX_WPMR
**Offset:** 0x01E4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key (Write-only)

| Value | Name | Description |
|---|---|---|
| 0x4D4154 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit.<br>Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See "Register Write Protection" for list of registers that can be write-protected.

| Value | Description |
|---|---|
| 0 | Disables the Write Protection if WPKEY corresponds to 0x4D4154 ("MAT" in ASCII). |
| 1 | Enables the Write Protection if WPKEY corresponds to 0x4D4154 ("MAT" in ASCII). |

### 18.13.11 Write Protection Status Register

**Name:** MATRIX_WPSR
**Offset:** 0x01E8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of MATRIX_WPSR. |
| 1 | A write protection violation has occurred since the last write of MATRIX_WPMR. |

### 18.13.12 Security Slave Registers

**Name:** MATRIX_SSRx
**Offset:** 0x0200 + x*0x04 [x=0..14]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WRNSECH7 | WRNSECH6 | WRNSECH5 | WRNSECH4 | WRNSECH3 | WRNSECH2 | WRNSECH1 | WRNSECH0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RDNSECH7 | RDNSECH6 | RDNSECH5 | RDNSECH4 | RDNSECH3 | RDNSECH2 | RDNSECH1 | RDNSECH0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LANSECH7 | LANSECH6 | LANSECH5 | LANSECH4 | LANSECH3 | LANSECH2 | LANSECH1 | LANSECH0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – WRNSECHx** Write Non-secured for HSELx Security Region Securable Area access rights:

| WRNSECHx / RDNSECHx | Non-secure Access | Secure Access |
|---|---|---|
| 00 | Denied | Read/Write |
| 01 | Read | Read/Write |
| 10 | Write | Read/Write |
| 11 | Read/Write | Read/Write |

| Value | Description |
|---|---|
| 0 | The HSELx AHB slave security region is split into one write secured and one write non-secured area, according to LANSECHx and MATRIX_SASSR.SASPLITx. That is, the so-defined secure high or low area is secured for Write access. |
| 1 | The HSELx AHB slave security region is non-secured for Write access. |

**Bits 8, 9, 10, 11, 12, 13, 14, 15 – RDNSECHx** Read Non-secured for HSELx Security Region

| Value | Description |
|---|---|
| 0 | The HSELx AHB slave security region is split into one read secured and one read non-secured area, according to LANSECHx and MATRIX_SASSR.SASPLITx. That is, the so-defined secure high or low area is secured for Read access. |
| 1 | The HSELx AHB slave security region is non-secure for Read access. |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – LANSECHx** Low Area Non-secure in HSELx Security Region

| Value | Description |
|---|---|
| 0 | The security of the HSELx AHB slave area lying below the corresponding MATRIX_SASSR.SASPLITx boundary is configured according to RDNSECHx and WRNSECHx. The entire remaining HSELx upper address space is configured as Non-secure access. |
| 1 | The HSELx AHB slave address area lying below the corresponding MATRIX_SASSR.SASPLITx boundary is configured as Non-secure access, and the entire remaining upper address space according to RDNSECHx and WRNSECHx. |

### 18.13.13 Security Areas Split Slave Registers

**Name:** MATRIX_SASSRx
**Offset:** 0x0240 + x*0x04 [x=0..14]
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SASPLIT7[3:0] | | | | SASPLIT6[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SASPLIT5[3:0] | | | | SASPLIT4[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SASPLIT3[3:0] | | | | SASPLIT2[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SASPLIT1[3:0] | | | | SASPLIT0[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

**Bits 0:3, 4:7, 8:11, 12:15, 16:19, 20:23, 24:27, 28:31 – SASPLITx** Security Areas Split for HSELx Security Region
This field defines the boundary address offset where the HSELx slave security region splits into two Security Areas with access controlled according to the corresponding MATRIX_SSR. It also defines the Security Low Area size inside the HSELx region.

If this Low Area size is set at or above the HSELx Region Size, then the Security High Area is no longer available and the MATRIX_SSR settings for the Low Area apply to the entire HSELx Security Region.

When applicable to a slave region, the initial value of MATRIX_SASSRx.SASPLITy is 0xF. When not applicable to a slave region, the initial value of MATRIX_SASSRx.SASPLITy is 0x0.

| SASPLITx | Split Offset | Security Low Area Size |
|---|---|---|
| 0000 | 0x00001000 | 4 Kbytes |
| 0001 | 0x00002000 | 8 Kbytes |
| 0010 | 0x00004000 | 16 Kbytes |
| 0011 | 0x00008000 | 32 Kbytes |
| 0100 | 0x00010000 | 64 Kbytes |
| 0101 | 0x00020000 | 128 Kbytes |

.....continued

| SASPLITx | Split Offset | Security Low Area Size |
|---|---|---|
| 0110 | 0x00040000 | 256 Kbytes |
| 0111 | 0x00080000 | 512 Kbytes |
| 1000 | 0x00100000 | 1 Mbyte |
| 1001 | 0x00200000 | 2 Mbytes |
| 1010 | 0x00400000 | 4 Mbytes |
| 1011 | 0x00800000 | 8 Mbytes |
| 1100 | 0x01000000 | 16 Mbytes |
| 1101 | 0x02000000 | 32 Mbytes |
| 1110 | 0x04000000 | 64 Mbytes |
| 1111 | 0x08000000 | 128 Mbytes |

### 18.13.14 Security Region Top Slave Registers

**Name:** MATRIX_SRTSRx
**Offset:** 0x0284 + x*0x04 [x=0..14]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SRTOP7[3:0] | | | | SRTOP6[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SRTOP5[3:0] | | | | SRTOP4[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SRTOP3[3:0] | | | | SRTOP2[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SRTOP1[3:0] | | | | SRTOP0[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0:3, 4:7, 8:11, 12:15, 16:19, 20:23, 24:27, 28:31 – SRTOPx** HSELx Security Region Top
This field defines the size of the HSELx security region address space. Invalid sizes for the slave region must never be programmed. Valid sizes and number of security regions are product-, slave- and slave-configuration dependent.

**Note:** The slaves featuring multiple scalable contiguous security regions have a single SRTOP0 field for all the security regions.

If this HSELx security region size is set at or below the HSELx low area size, then there is no Security High Area and the MATRIX_SSR settings for the Low Area apply to the whole HSELx security region.

| SRTOPx | Top Offset | Security Region Size |
|---|---|---|
| 0000 | 0x00001000 | 4 Kbytes |
| 0001 | 0x00002000 | 8 Kbytes |
| 0010 | 0x00004000 | 16 Kbytes |
| 0011 | 0x00008000 | 32 Kbytes |
| 0100 | 0x00010000 | 64 Kbytes |
| 0101 | 0x00020000 | 128 Kbytes |

| .....continued | | |
|---|---|---|
| **SRTOPx** | **Top Offset** | **Security Region Size** |
| 0110 | 0x00040000 | 256 Kbytes |
| 0111 | 0x00080000 | 512 Kbytes |
| 1000 | 0x00100000 | 1 Mbyte |
| 1001 | 0x00200000 | 2 Mbytes |
| 1010 | 0x00400000 | 4 Mbytes |
| 1011 | 0x00800000 | 8 Mbytes |
| 1100 | 0x01000000 | 16 Mbytes |
| 1101 | 0x02000000 | 32 Mbytes |
| 1110 | 0x04000000 | 64 Mbytes |
| 1111 | 0x08000000 | 128 Mbytes |

### 18.13.15 Security Peripheral Select x Registers

**Name:** MATRIX_SPSELRx
**Offset:** 0x02C0 + x*0x04 [x=0..2]
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

The actual number of peripherals implemented is device-specific; refer to the “Peripheral Identifiers” section for details.

Each MATRIX_SPSELR can configure the access security type for up to 32 peripherals:

- MATRIX_SPSELR1 configures the access security type for peripheral identifiers 0–31 (bits NSECP0–NSECP31).
- MATRIX_SPSELR2 configures the access security type for peripheral identifiers 32–63 (bits NSECP0–NSECP31).
- MATRIX_SPSELR3 configures the access security type for peripheral identifiers 64–95 (bits NSECP0–NSECP31).

Reset values are as follows:

- MATRIX_SPSELR1: 0x000D2504 for H32MX, 0xFFF2DAFB for H64MX
- MATRIX_SPSELR2: 0x011C0000 for H32MX, 0xFFE7FFFF for H64MX
- MATRIX_SPSELR3: 0xFFFFFFFA for H32MX, 0xFFFFFFE7 for H64MX

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NSECP31 | NSECP30 | NSECP29 | NSECP28 | NSECP27 | NSECP26 | NSECP25 | NSECP24 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NSECP23 | NSECP22 | NSECP21 | NSECP20 | NSECP19 | NSECP18 | NSECP17 | NSECP16 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NSECP15 | NSECP14 | NSECP13 | NSECP12 | NSECP11 | NSECP10 | NSECP9 | NSECP8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NSECP7 | NSECP6 | NSECP5 | NSECP4 | NSECP3 | NSECP2 | NSECP1 | NSECP0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – NSECPy** Non-secured Peripheral

| Value | Description |
|---|---|
| 0 | The selected peripheral address space is configured as “Secured” access (value of ‘0’ has no effect if the peripheral security type is “Peripheral Always Non-secured”). |

| Value | Description |
|---|---|
| 1 | The selected peripheral address space is configured as “Non-secured” access (value of ‘1’ has no effect if the peripheral security type is “Peripheral Always Secured”). |

# 19. Special Function Registers (SFR)

## 19.1 Description

Special Function Registers (SFR) manage specific aspects of the integrated memory, bridge implementations, processor and other functionality not controlled elsewhere.

## 19.2 Embedded Characteristics

- 32-bit Special Function Registers Control Specific Behavior of the Product

## 19.3 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x04 | SFR_DDRCFG | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | FDQSIEN | FDQIEN |
| | | 31:24 | | | | | | | | |
| 0x08 ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | SFR_OHCIICR | 7:0 | | | APPSTART | ARIE | | RES2 | RES1 | RES0 |
| | | 15:8 | | | | | | SUSPEND_C | SUSPEND_B | SUSPEND_A |
| | | 23:16 | UDPPUDIS | | | | | | | |
| | | 31:24 | | | | | HSIC_SEL | | | |
| 0x14 | SFR_OHCIISR | 7:0 | | | | | | | | RIS0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 ... 0x27 | Reserved | | | | | | | | | |
| 0x28 | SFR_SECURE | 7:0 | | | | | | | | ROM |
| | | 15:8 | | | | | | | | FUSE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x2C ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | SFR_UTMICKTRIM | 7:0 | | | | | | | FREQ[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | VBG[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x34 | SFR_UTMIHSTRIM | 7:0 | | DISC[2:0] | | | | SQUELCH[2:0] | | |
| | | 15:8 | | SLOPE1[2:0] | | | | SLOPE0[2:0] | | |
| | | 23:16 | | | | | | SLOPE2[2:0] | | |
| | | 31:24 | | | | | | | | |
| 0x38 | SFR_UTMIFSTRIM | 7:0 | | FALL[2:0] | | | | RISE[2:0] | | |
| | | 15:8 | | | | | | | XCVR[1:0] | |
| | | 23:16 | | ZP[2:0] | | | | ZN[2:0] | | |
| | | 31:24 | | | | | | | | |
| 0x3C | SFR_UTMISWAP | 7:0 | | | | | | PORT2 | PORT1 | PORT0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x40 ... 0x47 | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x48 | SFR_CAN | 7:0 | EXT_MEM_CAN0_ADDR[7:0] | | | | | | | |
| | | 15:8 | EXT_MEM_CAN0_ADDR[15:8] | | | | | | | |
| | | 23:16 | EXT_MEM_CAN1_ADDR[7:0] | | | | | | | |
| | | 31:24 | EXT_MEM_CAN1_ADDR[15:8] | | | | | | | |
| 0x4C | SFR_SN0 | 7:0 | SN0[7:0] | | | | | | | |
| | | 15:8 | SN0[15:8] | | | | | | | |
| | | 23:16 | SN0[23:16] | | | | | | | |
| | | 31:24 | SN0[31:24] | | | | | | | |
| 0x50 | SFR_SN1 | 7:0 | SN1[7:0] | | | | | | | |
| | | 15:8 | SN1[15:8] | | | | | | | |
| | | 23:16 | SN1[23:16] | | | | | | | |
| | | 31:24 | SN1[31:24] | | | | | | | |
| 0x54 | SFR_AICREDIR | 7:0 | AICREDIRKEY[6:0] | | | | | | | NSAIC |
| | | 15:8 | AICREDIRKEY[14:7] | | | | | | | |
| | | 23:16 | AICREDIRKEY[22:15] | | | | | | | |
| | | 31:24 | AICREDIRKEY[30:23] | | | | | | | |
| 0x58 | SFR_L2CC_HRAMC | 7:0 | | | | | | | | SRAM_SEL |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x5C ... 0x8F | Reserved | | | | | | | | | |
| 0x90 | SFR_I2SCLKSEL | 7:0 | | | | | | | CLKSEL1 | CLKSEL0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x94 | QSPICLK_REG | 7:0 | | | | | | | | SUP_SEL |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 19.3.1 DDR Configuration Register

**Name:** SFR_DDRCFG
**Offset:** 0x04
**Reset:** 0x00000001
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | FDQSIEN | FDQIEN |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 17 – FDQSIEN** Force DDR_DQS Input Buffer Always On
FDQSIEN = 1 is used to force the selection of the analog comparator inside the IO. If this bit is cleared, the DDR controller automatically manages the selection of the analog comparator. Forcing the bit to 0 reduces power consumption.

| Value | Description |
|---|---|
| 0 | DDR_DQS input buffer controlled by DDR controller. |
| 1 | DDR_DQS input buffer always on. |

**Bit 16 – FDQIEN** Force DDR_DQ Input Buffer Always On
FDQIEN = 1 is used to force the selection of the analog comparator inside the IO. If this bit is cleared, the DDR controller automatically manages the selection of the analog comparator. Forcing the bit to 0 reduces power consumption.

| Value | Description |
|---|---|
| 0 | DDR_DQ input buffer controlled by DDR controller. |
| 1 | DDR_DQ input buffer always on. |

### 19.3.2 OHCI Interrupt Configuration Register

**Name:** SFR_OHCIICR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | HSIC_SEL | | | |
| Access | | | | | R/W | | | |
| Reset | | | | | 0 | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UDPPUDIS | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SUSPEND_C | SUSPEND_B | SUSPEND_A |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | APPSTART | ARIE | | RES2 | RES1 | RES0 |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | 0 | 0 | | 0 | 0 | 0 |

**Bit 27 – HSIC_SEL** Reserved

| Value | Description |
|---|---|
| 0 | Must write 0. |

**Bit 23 – UDPPUDIS** USB DEVICE PULLUP DISABLE

| Value | Description |
|---|---|
| 0 | USB device pullup connection is enabled. |
| 1 | USB device pullup connection is disabled. |

**Bit 10 – SUSPEND_C** USB PORT C

| Value | Description |
|---|---|
| 0 | Suspends controlled by EHCI-OHCI. |
| 1 | Forces the suspend for PORTC. |

**Bit 9 – SUSPEND_B** USB PORT B

| Value | Description |
|---|---|
| 0 | Suspend controlled by EHCI-OHCI. |
| 1 | Forces the suspend for PORTB. |

**Bit 8 – SUSPEND_A** USB PORT A

| Value | Description |
|---|---|
| 0 | Suspends controlled by EHCI-OHCI. |
| 1 | Forces the suspend for PORTA. |

**Bit 5 – APPSTART** Reserved

| Value | Description |
|---|---|
| 0 | Must write 0. |

**Bit 4 – ARIE** OHCI Asynchronous Resume Interrupt Enable

| Value | Description |
|---|---|
| 0 | Interrupt disabled. |
| 1 | Interrupt enabled. |

**Bits 0, 1, 2 – RESx** USB PORTx RESET

| Value | Description |
|---|---|
| 0 | Resets USB Port. |
| 1 | Usable USB Port. |

### 19.3.3 OHCI Interrupt Status Register

**Name:** SFR_OHCIISR
**Offset:** 0x14
**Reset:** –
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | RIS0 |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | – |

**Bit 0 – RISx** OHCI Resume Interrupt Status Port x

| Value | Description |
|---|---|
| 0 | OHCI port resume not detected. |
| 1 | OHCI port resume detected. |

### 19.3.4 Security Configuration Register

**Name:** SFR_SECURE
**Offset:** 0x28
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | FUSE |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ROM |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 8 – FUSE** Disable Access to Fuse Controller
This bit is writable once only. When the Fuse Controller is secured, only a reset signal can clear this bit.

| Value | Description |
|---|---|
| 0 | Fuse Controller is enabled. |
| 1 | Fuse Controller is disabled. |

**Bit 0 – ROM** Disable Access to ROM Code
This bit is writable once only. When the ROM is secured, only a reset signal can clear this bit.

| Value | Description |
|---|---|
| 0 | ROM is enabled. |
| 1 | ROM is disabled. |

### 19.3.5 UTMI Clock Trimming Register

**Name:** SFR_UTMICKTRIM
**Offset:** 0x30
**Reset:** 0x00010000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | VBG[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | FREQ[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bits 17:16 – VBG[1:0]** UTMI Band Gap Voltage Trimming

**Bits 1:0 – FREQ[1:0]** UTMI Reference Clock Frequency

| Value | Name | Description |
|---|---|---|
| 0 | 12 | 12 MHz reference clock |
| 1 | 16 | 16 MHz reference clock |
| 2 | 24 | 24 MHz reference clock |
| 3 | 12 | 12 MHz reference clock |

### 19.3.6 UTMI High-Speed Trimming Register

**Name:** SFR_UTMIHSTRIM
**Offset:** 0x34
**Reset:** 0x00044433
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SLOPE2[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 1 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | SLOPE1[2:0] | | | | SLOPE0[2:0] | | |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 1 | 0 | 0 | | 1 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | DISC[2:0] | | | | SQUELCH[2:0] | | |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 1 | 1 | | 0 | 1 | 1 |

**Bits 8:10, 12:14, 16:18 – SLOPEx** UTMI HS PORTx Transceiver Slope Trimming
Calibration bits to adjust HS Transceiver output slope for PORTx.

**Bits 6:4 – DISC[2:0]** UTMI Disconnect Voltage Trimming
Calibration bits to adjust disconnect threshold.

**Bits 2:0 – SQUELCH[2:0]** UTMI HS SQUELCH Voltage Trimming
Calibration bits to adjust squelch threshold.

### 19.3.7 UTMI Full-Speed Trimming Register

**Name:** SFR_UTMIFSTRIM
**Offset:** 0x38
**Reset:** 0x00430211
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | ZP[2:0] | | | | ZN[2:0] | | |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 1 | 0 | 0 | | 0 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | XCVR[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 1 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | FALL[2:0] | | | | RISE[2:0] | | |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 0 | 1 | | 0 | 0 | 1 |

**Bits 22:20 – ZP[2:0]** FS Transceiver PMOS Impedance Trimming
Calibration bits to adjust the FS transceiver PMOS output impedance.

**Bits 18:16 – ZN[2:0]** FS Transceiver NMOS Impedance Trimming
Calibration bits to adjust the FS transceiver NMOS output impedance.

**Bits 9:8 – XCVR[1:0]** FS Transceiver Crossover Voltage Trimming
Calibration bits to adjust the FS transceiver crossover voltage.

**Bits 6:4 – FALL[2:0]** FS Transceiver Output Falling Slope Trimming
Calibration bits to adjust the FS transceiver output falling slope.

**Bits 2:0 – RISE[2:0]** FS Transceiver Output Rising Slope Trimming
Calibration bits to adjust the FS transceiver output rising slope.

### 19.3.8 UMTI DP/DM Pin Swapping Register

**Name:** SFR_UTMISWAP
**Offset:** 0x3C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PORT2 | PORT1 | PORT0 |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 0, 1, 2 – PORTx** PORT x DP/DM Pin Swapping
0 (NORMAL): DP/DM normal pinout.

1 (SWAPPED): DP/DM swapped pinout.

### 19.3.9 CAN Memories Address-based Register

**Name:** SFR_CAN
**Offset:** 0x48
**Reset:** 0x00200020
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | EXT_MEM_CAN1_ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | EXT_MEM_CAN1_ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EXT_MEM_CAN0_ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EXT_MEM_CAN0_ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – EXT_MEM_CAN1_ADDR[15:0]** MSB CAN1 DMA Base Address
Gives the 16-bit MSB of the CAN1 DMA base address. The 16-bit LSB must be programmed in the CAN1 user interface.

**Bits 15:0 – EXT_MEM_CAN0_ADDR[15:0]** MSB CAN0 DMA Base Address
Gives the 16-bit MSB of the CAN0 DMA base address. The 16-bit LSB must be programmed in the CAN0 user interface.

### 19.3.10 Serial Number 0 Register

**Name:** SFR_SN0
**Offset:** 0x4C
**Reset:** –
**Property:** Read-only

This register is used to read the first 32 bits of the 64-bit Serial Number (unique ID).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SN0[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SN0[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SN0[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SN0[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

**Bits 31:0 – SN0[31:0]** Serial Number 0

### 19.3.11 Serial Number 1 Register

**Name:** SFR_SN1
**Offset:** 0x50
**Reset:** –
**Property:** Read-only

This register is used to read the last 32 bits of the 64-bit Serial Number (unique ID).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SN1[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SN1[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SN1[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SN1[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

**Bits 31:0 – SN1[31:0]** Serial Number 1

### 19.3.12 AIC Interrupt Redirection Register

**Name:** SFR_AICREDIR
**Offset:** 0x54
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | AICREDIRKEY[30:23] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | AICREDIRKEY[22:15] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | AICREDIRKEY[14:7] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | AICREDIRKEY[6:0] | | | | | | | NSAIC |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:1 – AICREDIRKEY[30:0]** Unlock Key
Value is a XOR between 0xb6d81c4d and SN1[31:0] but only field [31:1] of the result must be written in this field. In case of set in Secure mode by fuse configuration, this register is read_only 0 (it is not possible to redirect secure interrupts on non-secure AIC for products set in secure mode for security reasons).

After three tries, entering a wrong key results in locking the NSAIC bit. A reset is needed.

**Bit 0 – NSAIC** Interrupt Redirection to Non-Secure AIC

| Value | Description |
|---|---|
| 0 | Interrupts are managed by the AIC corresponding to the Secure State of the peripheral (secure AIC or non-secure AIC). |
| 1 | All interrupts are managed by the non-secure AIC. |

### 19.3.13 HRAMC L2CC Register

**Name:** SFR_L2CC_HRAMC
**Offset:** 0x58
**Reset:** 0x00000000
**Property:** Read/Write

This register is used to configure the L2 cache to be used as an internal SRAM.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SRAM_SEL |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – SRAM_SEL** SRAM Selector

| Value | Description |
|---|---|
| 0 | Selects SRAM. |
| 1 | Selects L2CC. |

### 19.3.14 I2S Register

**Name:** SFR_I2SCLKSEL
**Offset:** 0x90
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CLKSEL1 | CLKSEL0 |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – CLKSEL1** Clock Selection 1

| Value | Description |
|---|---|
| 0 | Selects PCLK (peripheral clock). |
| 1 | Selects GCLK. |

**Bit 0 – CLKSEL0** Clock Selection 0

| Value | Description |
|---|---|
| 0 | Selects PCLK (peripheral clock). |
| 1 | Selects GCLK. |

### 19.3.15 QSPI Clock Pad Supply Select Register

**Name:** QSPICLK_REG
**Offset:** 0x94
**Reset:** 0x1
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SUP_SEL |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 1 |

**Bit 0 – SUP_SEL** Supply Selection

| Value | Description |
|---|---|
| 0 | 1.8V supply selected. |
| 1 | 3.3V supply selected. |

# 20. Special Function Registers Backup (SFRBU)

## 20.1 Description

Special Function Registers Backup (SFRBU) manages specific aspects of the integrated memory, bridge implementations, processor and other functionality not controlled elsewhere.

## 20.2 Embedded Characteristics

- 32-bit Special Function Registers Backup Controls Specific Behavior of the Product

## 20.3 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | SFRBU_PSWBUCTRL | 7:0 | | | | | STATE | SMCTRL | SSWCTRL | SCTRL |
| | | 15:8 | KEY_PSW_MODE[7:0] | | | | | | | |
| | | 23:16 | KEY_PSW_MODE[15:8] | | | | | | | |
| | | 31:24 | KEY_PSW_MODE[23:16] | | | | | | | |
| 0x04 | SFRBU_TSRANGECFG | 7:0 | | | | | | | | TSHRSEL |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | SFRBU_DDRBUMCR | 7:0 | | | | | | | | BUMEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | SFRBU_RXLPPUCR | 7:0 | | | | | | | | RXDPUCTRL |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 20.3.1 SFRBU Power Switch BU Control Register

**Name:** SFRBU_PSWBUCTRL
**Offset:** 0x00
**Reset:** 0x09
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY_PSW_MODE[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEY_PSW_MODE[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | KEY_PSW_MODE[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | STATE | SMCTRL | SSWCTRL | SCTRL |
| Access | | | | | R | R/W | R/W | R/W |
| Reset | | | | | 1 | 0 | 0 | 1 |

**Bits 31:8 – KEY_PSW_MODE[23:0]** Specific value mandatory to allow writing of other register bits (Write-only)
This field is a security key to prevent power switch changes due to software error or malicious code.

| Value | Description |
|---|---|
| 0x4BD20C | SFRBU_PSWBUCTRL register write possible. |
| Other values | SFRBU_PSWBUCTRL register write impossible. |

**Bit 3 – STATE** Power Switch BU state (Read-only)
Reflects the power switch BU supply source selection in real time. After a switching request, the user must wait for the analog cell switching time to have an updated status (see the section "Electrical Characteristics").

| Value | Description |
|---|---|
| 0 | LDO BU Supply source is VDDBU. |
| 1 | LDO BU Supply source is VDDANA. |

**Bit 2 – SMCTRL** Allow Power Switch BU Control by Security Module Autobackup (Hardware)
Enables automatic selection of the VDDBU source when the security module enters Backup mode.

This automatic supply source switching is independent from the SCTRL and SSWCTRL bits.

| Value | Description |
|---|---|
| 0 | Reset value. No automatic supply source switching from security module. |
| 1 | Automatic supply source switching from security module activated. |

**Bit 1 – SSWCTRL** Power Switch BU Source Selection
Has an action only if SCTRL bit value is “1”.

| Value | Description |
|---|---|
| 0 | Reset value. LDO Supply source is VDDBU. |
| 1 | LDO Supply source is VDDANA. |

**Bit 0 – SCTRL** Power Switch BU Software Control
Used to control the Power Switch BU state by software in addition to the SSWCTRL bit.

| Value | Description |
|---|---|
| 0 | Power Switch BU is controlled by hardware (SSWCTRL bit has no action). |
| 1 | Reset value. Power Switch BU is controlled by software (SSWCTRL bit has an action). |

### 20.3.2 SFRBU Temperature Sensor Range Configuration Register

**Name:** SFRBU_TSRANGECFG
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | TSHRSEL |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – TSHRSEL** Temperature Sensor Range Selection

| Value | Description |
|---|---|
| 0 | Reset value. Temperature sensor high triggering level is +105°C (internal transistor junction temperature). |
| 1 | Temperature sensor high triggering level is +115°C (internal transistor junction temperature). |

### 20.3.3 SFRBU DDR BU Mode Control Register

**Name:** SFRBU_DDRBUMCR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | BUMEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – BUMEN** DDR BU Mode Enable
Isolates the DDR pads from the CPU domain (VDDCORE).

Must be set after enabling the Self-refresh mode on the DDR memory and before powering down on VDDCORE.

To enable Self-refresh mode, refer to the MPDDRC Low-power register (MPDDRC_LPR) in the section "Multi-port DDR-SDRAM Controller" and to "Backup Mode with DDR in Self-refresh" in the section "Electrical Characteristics".

| Value | Description |
|---|---|
| 0 | Reset value. DDR Backup mode disabled. The DDR pads are not isolated from CPU domain. |
| 1 | DDR Backup mode enabled. The DDR pads are isolated from CPU domain (IOs are in memory state). |

### 20.3.4 SFRBU RXLP Pull-Up Control Register

**Name:** SFRBU_RXLPPUCR
**Offset:** 0x14
**Reset:** 0x01
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | RXDPUCTRL |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 1 |

**Bit 0 – RXDPUCTRL** RXLP RXD Pull-Up Control
If the RXLP is not used, it is recommended to clear this bit (enable the pull-up) to avoid power consumption on the VDDBU rail.

| Value | Description |
|---|---|
| 0 | Reset value. Pull-up enabled on RXD IO. |
| 1 | Pull-up disabled on RXD IO. |

# 21. Advanced Interrupt Controller (AIC)

## 21.1 Description

The Advanced Interrupt Controller (AIC) is an 8-level priority, individually maskable, vectored interrupt controller providing handling of up to one hundred and twenty-eight interrupt sources. It is designed to substantially reduce the software and real-time overhead in handling internal and external interrupts.

The AIC drives the nFIQ (fast interrupt request) and the nIRQ (standard interrupt request) inputs of an ARM processor. Inputs of the AIC are either internal peripheral interrupts or external interrupts coming from the product's pins.

The 8-level Priority Controller allows the user to define the priority for each interrupt source, thus permitting higher priority interrupts to be serviced even if a lower priority interrupt is being processed.

Internal interrupt sources can be programmed to be level-sensitive or edge-triggered. External interrupt sources can be programmed to be rising-edge or falling-edge triggered or high-level or low-level sensitive.

## 21.2 Embedded Characteristics

- Controls the Interrupt Lines (nIRQ and nFIQ) of an ARM Processor
- 128 Individually Maskable and Vectored Interrupt Sources
  - Source 0 is reserved for the fast interrupt input (FIQ)
  - Source 74 is reserved for system peripheral interrupts
  - Sources 2 to 73 and Sources 75 to 127 control up to 125 embedded peripheral interrupts or external interrupts
  - Programmable edge-triggered or level-sensitive internal sources
  - Programmable rising/falling edge-triggered or high/low level-sensitive external sources
- 8-level Priority Controller
  - Drives the normal interrupt of the processor
  - Handles priority of the interrupt sources 1 to 127
  - Higher priority interrupts can be served during service of lower priority interrupt
- Vectoring
  - Optimizes interrupt service routine branch and execution
  - One 32-bit vector register for all interrupt sources
  - Interrupt vector register reads the corresponding current interrupt vector
- Protect Mode
  - Easy debugging by preventing automatic operations when protect models are enabled
- General Interrupt Mask
  - Provides processor synchronization on events without triggering an interrupt
- Register Write Protection
- AIC0 is Non-Secure AIC, AIC1 is Secure AIC
- AIC0 manages nIRQ line, AIC1 manages nFIQ line

## 21.3 Block Diagram

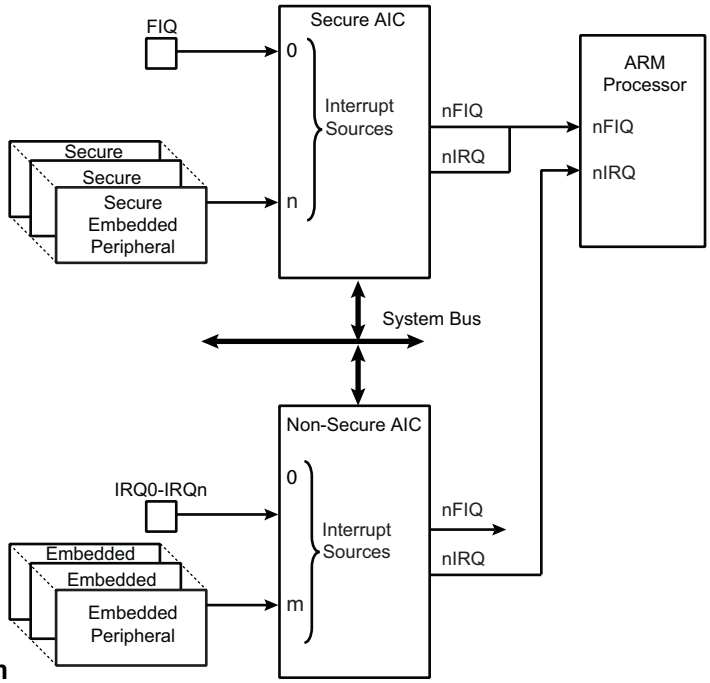
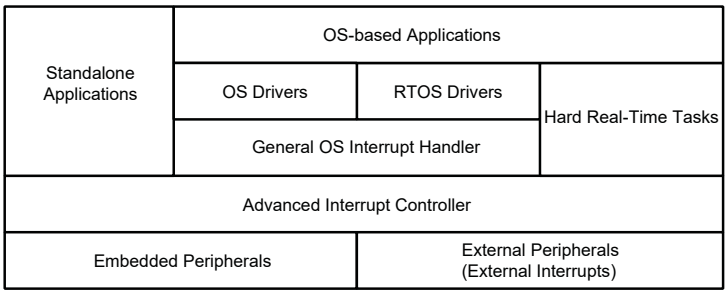



**Figure 21-1. Block Diagram**

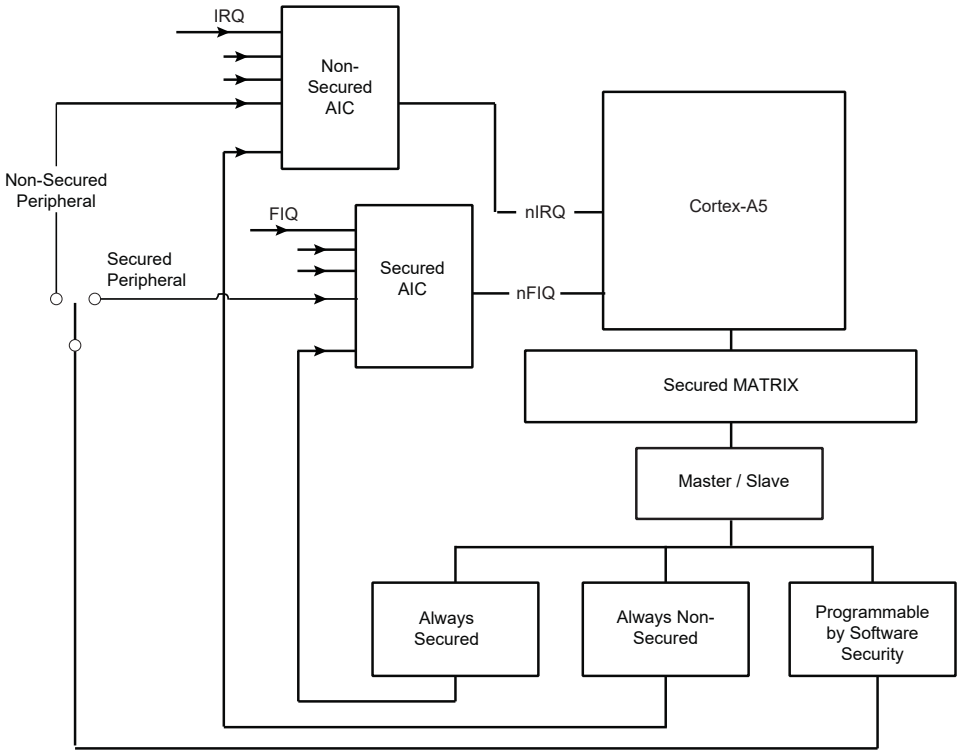
## 21.4 Application Block Diagram

**Figure 21-2. Description of the Application Block**



## 21.5 AIC Detailed Block Diagram

**Figure 21-3. AIC Detailed Block Diagram**




## 21.6 I/O Line Description

**Table 21-1. I/O Line Description**

| Pin Name | Pin Description | Type |
|---|---|---|
| FIQ | Fast Interrupt | Input |
| IRQ0–IRQn | Interrupt 0–Interrupt n | Input |

## 21.7 Product Dependencies

### 21.7.1 I/O Lines

The interrupt signals FIQ and IRQ0 to IRQn are normally multiplexed through the PIO controllers. Depending on the features of the PIO controller used in the product, the pins must be programmed in accordance with their assigned interrupt functions. This is not applicable when the PIO controller used in the product is transparent on the input path.

### 21.7.2 Power Management

The AIC is continuously clocked. The Power Management Controller has no effect on the AIC behavior.

The assertion of the AIC outputs, either nIRQ or nFIQ, wakes up the ARM processor while it is in Idle mode. The General Interrupt Mask feature enables the AIC to wake up the processor without asserting the interrupt line of the processor, thus providing synchronization of the processor on an event.

### 21.7.3 Interrupt Sources

FIQ always drives Interrupt Source 0.

The System Controller interrupt drives Interrupt Source 74.

The System Controller interrupt is the result of the OR-wiring of the System Controller interrupt lines. When a System Controller interrupt occurs, the service routine must first distinguish the cause of the interrupt. This is performed by reading successively the status registers of the System Controller peripherals.

Interrupt sources 2 to 73 and 75 to 127 can either be connected to the interrupt outputs of an embedded user peripheral, or to external interrupt lines. The external interrupt lines can be connected either directly or through the PIO Controller.

PIO controllers are considered as user peripherals in the scope of interrupt handling. Accordingly, the PIO controller interrupt lines are connected to interrupt sources 2 to 73 and 75 to 127.

The peripheral identification defined at the product level corresponds to the interrupt source number (as well as the bit number controlling the clock of the peripheral). Consequently, to simplify the description of the functional operations and the user interface, the interrupt sources are named FIQ, SYS, and PID2 to PID73 and PID75 to PID127.

AIC0 manages all Non-Secure Interrupts including IRQn; AIC1 manages all Secure Interrupts including FIQ.

Each AIC has its own User Interface. The user should pay attention to use the relevant user interface for each source.

## 21.8 Functional Description

### 21.8.1 Interrupt Source Control

#### 21.8.1.1 Interrupt Source Mode

The AIC independently programs each interrupt source. The SRCTYPE field of the Source Mode register (AIC_SMR) selects the interrupt condition of the interrupt source selected by the INTSEL field of the Source Select register (AIC_SSR).

**Note:** Configuration registers such as AIC_SMR and AIC_SSR return the values corresponding to the interrupt source selected by INTSEL.

The internal interrupt sources wired on the interrupt outputs of the embedded peripherals can be programmed either in Level-Sensitive mode or in Edge-Triggered mode. The active level of the internal interrupts is not important for the user.

The external interrupt sources can be programmed either in High Level-Sensitive or Low Level-Sensitive modes, or in Rising Edge-Triggered or Negative Edge-Triggered modes.

#### 21.8.1.2 Interrupt Source Enabling

Each interrupt source, including the FIQ in source 0, can be enabled or disabled by using the command registers Interrupt Enable Command register (AIC_IECR) and Interrupt Disable Command register (AIC_IDCR). The interrupt mask of the selected interrupt source can be read in the Interrupt Mask register (AIC_IMR). A disabled interrupt does not affect servicing of other interrupts.

#### 21.8.1.3 Interrupt Clearing and Setting

All interrupt sources programmed to be edge-triggered (including the FIQ in source 0) can be individually set or cleared by writing respectively the Interrupt Set Command register (AIC_ISCR) and Interrupt Clear

Command register (AIC_ICCR). Clearing or setting interrupt sources programmed in Level-Sensitive mode has no effect.

The clear operation is perfunctory, as the software must perform an action to reset the “memorization” circuitry activated when the source is programmed in Edge-Triggered mode. However, the set operation is available for auto-test or software debug purposes. It can also be used to execute an AIC-implementation of a software interrupt.

The AIC features an automatic clear of the current interrupt when AIC_IVR (Interrupt Vector register) is read. Only the interrupt source being detected by the AIC as the current interrupt is affected by this operation. (See the section “Priority Controller”.) The automatic clear reduces the operations required by the interrupt service routine entry code to read AIC_IVR.

The automatic clear of interrupt source 0 is performed when the FIQ Vector register (AIC_FVR) is read.

#### 21.8.1.4 Interrupt Status

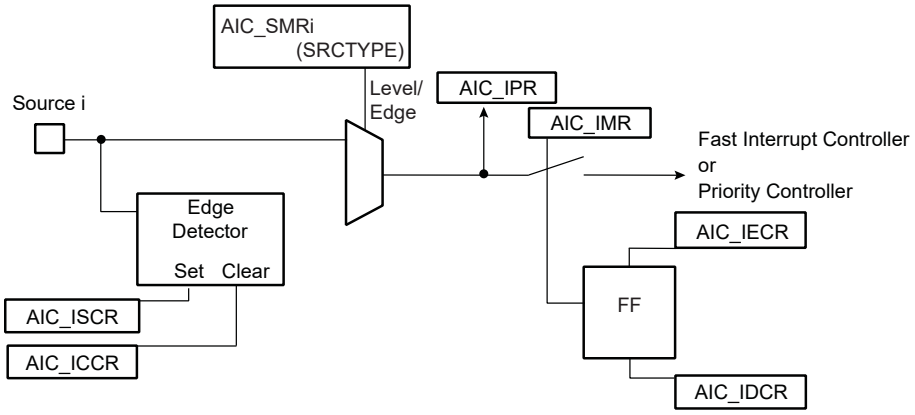
Interrupt Pending registers (AIC_IPR) represent the state of the interrupt lines, whether they are masked or not. AIC_IMR can be used to define the mask of the interrupt lines.

The Interrupt Status register (AIC_ISR) reads the number of the current interrupt (see the section ”Priority Controller”) and the Core Interrupt Status register (AIC_CISR) gives an image of the nIRQ and nFIQ signals driven on the processor.

Each status referred to above can be used to optimize the interrupt handling of the systems.

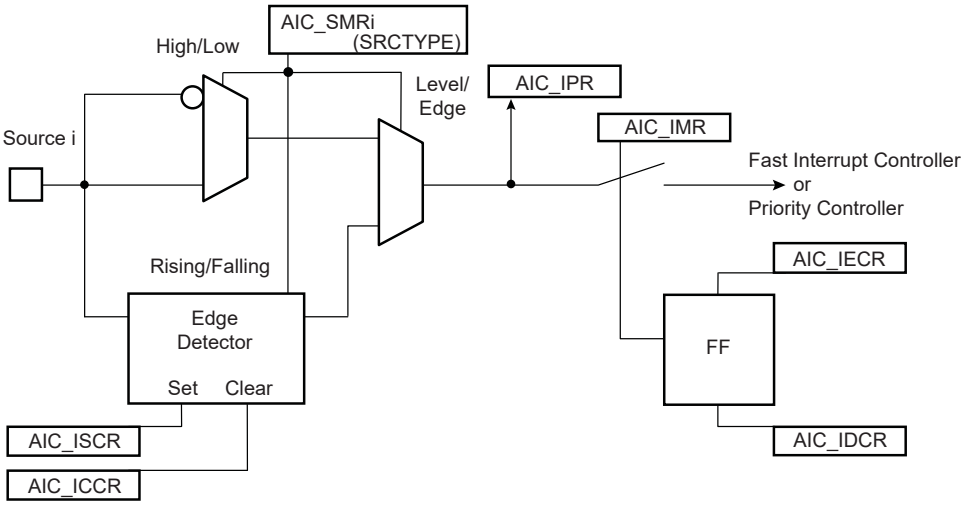
#### 21.8.1.5 Internal Interrupt Source Input Stage

**Figure 21-4. Internal Interrupt Source Input Stage**



#### 21.8.1.6 External Interrupt Source Input Stage

**Figure 21-5. External Interrupt Source Input Stage**




### 21.8.2 Interrupt Latencies

Global interrupt latencies depend on several parameters, including:

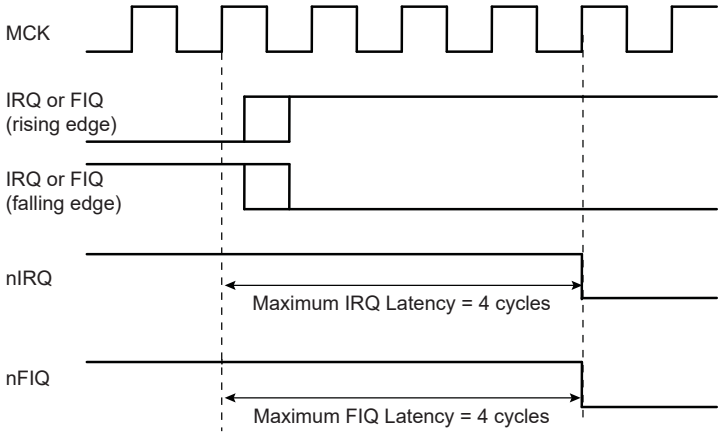
- The time the software masks the interrupts
- Occurrence, either at the processor level or at the AIC level
- The execution time of the instruction in progress when the interrupt occurs
- The treatment of higher priority interrupts and the resynchronization of the hardware signals

This section addresses hardware resynchronizations only. It gives details about the latency times between the events on an external interrupt leading to a valid interrupt (edge or level) or the assertion of an internal interrupt source and the assertion of the nIRQ or nFIQ line on the processor. The resynchronization time depends on the programming of the interrupt source and on its type (internal or external). For the standard interrupt, resynchronization times are given assuming there is no higher priority in progress.

The PIO Controller multiplexing has no effect on the interrupt latencies of the external interrupt sources.

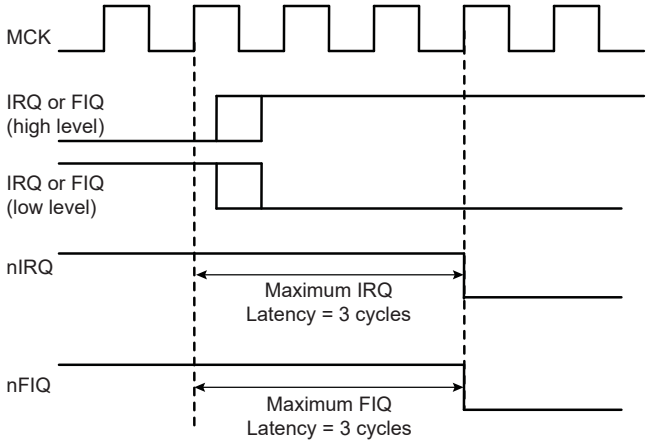
#### 21.8.2.1 External Interrupt Edge Triggered Source

**Figure 21-6. External Interrupt Edge Triggered Source**



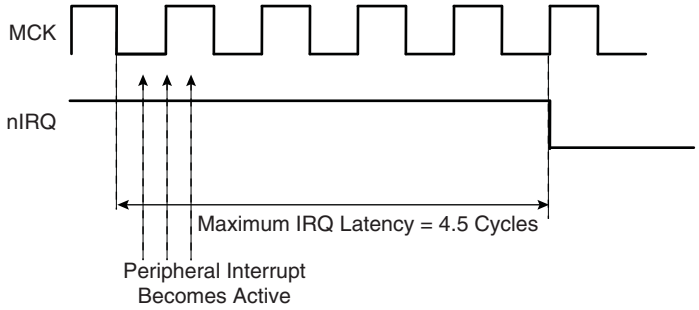
#### 21.8.2.2 External Interrupt Level Sensitive Source

**Figure 21-7. External Interrupt Level Sensitive Source**



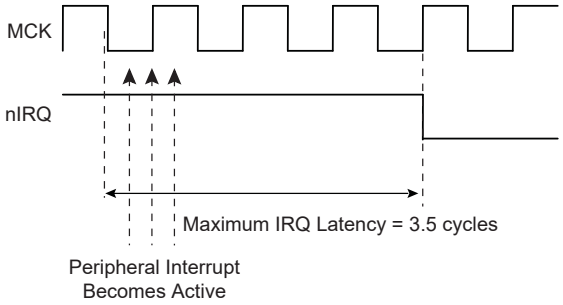

#### 21.8.2.3 Internal Interrupt Edge Triggered Source

**Figure 21-8. Internal Interrupt Edge Triggered Source**




#### 21.8.2.4 Internal Interrupt Level Sensitive Source

**Figure 21-9. Internal Interrupt Level Sensitive Source**




### 21.8.3 Normal Interrupt

#### 21.8.3.1 Priority Controller

An 8-level priority controller drives the nIRQ line of the processor, depending on the interrupt conditions occurring on the interrupt sources 1 to 127.

Each interrupt source has a programmable priority level of 7 to 0, which is user-definable by writing AIC_SMR.PRIOR. Level 7 is the highest priority and level 0 the lowest.

As soon as an interrupt condition occurs, as defined by AIC_SMR.SRCTYPE, the nIRQ line is asserted. As a new interrupt condition might have happened on other interrupt sources since the nIRQ has been asserted, the priority controller determines the current interrupt at the time AIC_IVR is read. The read of

AIC_IVR is the entry point of the interrupt handling which allows the AIC to consider that the interrupt has been taken into account by the software.

The current priority level is defined as the priority level of the current interrupt.

If several interrupt sources of equal priority are pending and enabled when AIC_IVR is read, the interrupt with the lowest interrupt source number is serviced first.

The nIRQ line can be asserted only if an interrupt condition occurs on an interrupt source with a higher priority. If an interrupt condition happens (or is pending) during the interrupt treatment in progress, it is delayed until the software indicates to the AIC the end of the current service by writing AIC_EOICR (End of Interrupt Command register). The write of AIC_EOICR is the exit point of the interrupt handling.

#### 21.8.3.2 Interrupt Nesting

The priority controller utilizes interrupt nesting in order for the high priority interrupt to be handled during the service of lower priority interrupts. This requires the interrupt service routines of the lower interrupts to re-enable the interrupt at the processor level.

When an interrupt of a higher priority happens during an already occurring interrupt service routine, the nIRQ line is re-asserted. If the interrupt is enabled at the core level, the current execution is interrupted and the new interrupt service routine should read AIC_IVR. At this time, the current interrupt number and its priority level are pushed into an embedded hardware stack, so that they are saved and restored when the higher priority interrupt servicing is finished and AIC_EOICR is written.

The AIC is equipped with an 8-level wide hardware stack in order to support up to eight interrupt nestings to match the eight priority levels.

#### 21.8.3.3 Interrupt Handlers

This section gives an overview of the fast interrupt handling sequence when using the AIC. It is assumed that the programmer understands the architecture of the ARM processor, and especially the Processor Interrupt modes and the associated status bits.

It is assumed that:

1. The AIC has been programmed, AIC_SVR registers are loaded with corresponding interrupt service routine addresses and interrupts are enabled.
2. The instruction at the ARM interrupt exception vector address is required to work with the vectoring. Load the PC with the absolute address of the interrupt handler.

When nIRQ is asserted, if the bit “I” of CPSR is 0, the sequence is as follows:

1. The CPSR is stored in SPSR_irq, the current value of the Program Counter is loaded in the Interrupt link register (R14_irq) and the Program Counter (R15) is loaded with 0x18. In the following cycle during fetch at address 0x1C, the ARM core adjusts R14_irq, decrementing it by four.
2. The ARM core enters Interrupt mode, if it has not already done so.
3. When the instruction loaded at address 0x18 is executed, the program counter is loaded with the value read in AIC_IVR. Reading AIC_IVR has the following effects:
   - Sets the current interrupt to be the pending and enabled interrupt with the highest priority. The current level is the priority level of the current interrupt.
   - De-asserts the nIRQ line on the processor. Even if vectoring is not used, AIC_IVR must be read in order to de-assert nIRQ.
   - Automatically clears the interrupt, if it has been programmed to be edge-triggered.
   - Pushes the current level and the current interrupt number on to the stack.
   - Returns the value written in AIC_SVR corresponding to the current interrupt.

4. The previous step has the effect of branching to the corresponding interrupt service routine. This should start by saving the link register (R14_irq) and SPSR_IRQ. The link register must be decremented by four when it is saved if it is to be restored directly into the program counter at the end of the interrupt. For example, the instruction `SUB PC, LR, #4` may be used.
5. Further interrupts can then be unmasked by clearing the “I” bit in CPSR, allowing re-assertion of the nIRQ to be taken into account by the core. This can happen if an interrupt with a higher priority than the current interrupt occurs.
6. The interrupt handler can then proceed as required, saving the registers that will be used and restoring them at the end. During this phase, an interrupt of higher priority than the current level will restart the sequence from step 1.
   **Note:** If the interrupt is programmed to be level-sensitive, the source of the interrupt must be cleared during this phase.
7. The “I” bit in CPSR must be set in order to mask interrupts before exiting to ensure that the interrupt is completed in an orderly manner.
8. AIC_EOICR must be written in order to indicate to the AIC that the current interrupt is finished. This causes the current level to be popped from the stack, restoring the previous current level if one exists on the stack. If another interrupt is pending, with lower or equal priority than the old current level but with higher priority than the new current level, the nIRQ line is re-asserted, but the interrupt sequence does not immediately start because the “I” bit is set in the core. SPSR_irq is restored. Finally, the saved value of the link register is restored directly into the PC. This has the effect of returning from the interrupt to whatever was being executed before, and of loading the CPSR with the stored SPSR, masking or unmasking the interrupts depending on the state saved in SPSR_irq.
   **Note:** The “I” bit in SPSR is significant. If it is set, it indicates that the ARM core was on the verge of masking an interrupt when the mask instruction was interrupted. Hence, when SPSR is restored, the mask instruction is completed (interrupt is masked).

### 21.8.4 Fast Interrupt

#### 21.8.4.1 Fast Interrupt Source

Interrupt source 0 is the only source which can raise a fast interrupt request to the processor. Interrupt source 0 is generally connected to a FIQ pin of the product, either directly or through a PIO Controller.

#### 21.8.4.2 Fast Interrupt Control

The fast interrupt logic of the AIC has no priority controller. The mode of interrupt source 0 is programmed with AIC_SMR and INTSEL = 0; the PRIOR field of this register is not used even if it reads what has been written. AIC_SMR.SRCTYPE enables programming the fast interrupt source to be rising-edge triggered or falling-edge triggered or high-level sensitive or low-level sensitive.

Writing 0x1 in AIC_IECR and AIC_IDCR respectively enables and disables the fast interrupt when INTSEL = 0. Bit 0 of AIC_IMR indicates whether the fast interrupt is enabled or disabled.

#### 21.8.4.3 Fast Interrupt Handlers

This section gives an overview of the fast interrupt handling sequence when using the AIC. It is assumed that the programmer understands the architecture of the ARM processor, and especially the Processor Interrupt modes and associated status bits.

Assuming that:

1. The AIC has been programmed, AIC_SVR is loaded with the fast interrupt service routine address, and interrupt source 0 is enabled.

2. The Instruction at address 0x1C (FIQ exception vector address) is required to vector the fast interrupt. Load the PC with the absolute address of the interrupt handler.
3. The user does not need nested fast interrupts.

When nFIQ is asserted, if bit “F” of CPSR is 0, the sequence is:

1. The CPSR is stored in SPSR_fiq, the current value of the program counter is loaded in the FIQ link register (R14_FIQ) and the program counter (R15) is loaded with 0x1C. In the following cycle, during fetch at address 0x20, the ARM core adjusts R14_fiq, decrementing it by four.
2. The ARM core enters FIQ mode.
3. The routine must read AIC1_CISR to know if the interrupt is the FIQ or a Secure Internal interrupt.

```
   ldr r1, =REG_SAIC_CISR
   ldr r1, [r1]
   cmp r1, #AIC_CISR_NFIQ
   beq get_fiqvec_addr
```

   If FIQ is active, it is processed in priority, even if another interrupt is active.

```
get_irqvec_addr
   ldr r14, =REG_SAIC_IVR
   b read_vec
get_fiqvec_addr
   ldr r14, =REG_SAIC_FVR
read_vec
   ldr r0, [r14]
```

   Now r0 contains the correct vector address, IVR for a Secure Internal interrupt or FVR for FIQ.

   The system can branch to the routine pointed to by r0.

```
FIQ_Handler_Branch
   mov r14, pc
   bx r0
```

4. The previous step enables branching to the corresponding interrupt service routine. It is not necessary to save the link register R14_fiq and SPSR_fiq if nested fast interrupts are not needed.
5. The Interrupt Handler can then proceed as required. It is not necessary to save registers R8 to R13 because the FIQ mode has its own dedicated registers and registers R8 to R13 are banked. The other registers, R0 to R7, must be saved before being used, and restored at the end (before the next step).
   **Note:** If the fast interrupt is programmed to be level-sensitive, the source of the interrupt must be cleared during this phase in order to de-assert interrupt source 0.
6. Finally, Link register R14_fiq is restored into the PC after decrementing it by four (with instruction `SUB PC, LR, #4` for example). This has the effect of returning from the interrupt to whatever was being executed before, loading the CPSR with the SPSR and masking or unmasking the fast interrupt depending on the state saved in the SPSR.
   **Note:** The “F” bit in SPSR is significant. If it is set, it indicates that the ARM core was just about to mask FIQ interrupts when the mask instruction was interrupted. Hence, when the SPSR is restored, the interrupted instruction is completed (FIQ is masked).

Another way to handle the fast interrupt is to map the interrupt service routine at the address of the ARM vector 0x1C. This method does not use vectoring, so that reading AIC_FVR must be performed at the very beginning of the handler operation. However, this method saves the execution of a branch instruction.

### 21.8.5 Protect Mode

The Protect mode is used to read the Interrupt Vector register without performing the associated automatic operations. This is necessary when working with a debug system. When a debugger, working either with a Debug Monitor or the ARM processor's ICE, stops the applications and updates the opened windows, it might read the AIC User Interface and thus the IVR. This has adverse consequences:

- If an enabled interrupt with a higher priority than the current one is pending, it is stacked.
- If there is no enabled pending interrupt, the spurious vector is returned.

In either case, an End of Interrupt command is necessary to acknowledge and restore the context of the AIC. This operation is generally not performed by the debug system, as the debug system would become strongly intrusive and cause the application to enter an undesired state.

This is avoided by using the Protect mode. Writing PROT in the Debug Control register (AIC_DCR) at 0x1 enables the Protect mode.

When the Protect mode is enabled, the AIC performs interrupt stacking only when a write access is performed on AIC_IVR. Therefore, the Interrupt Service Routines must write (arbitrary data) to AIC_IVR just after reading it. The new context of the AIC, including the value of AIC_ISR, is updated with the current interrupt only when AIC_IVR is written.

An AIC_IVR read on its own (e.g., by a debugger) modifies neither the AIC context nor AIC_ISR. Extra AIC_IVR reads perform the same operations. However, it is recommended to not stop the processor between the read and the write of AIC_IVR of the interrupt service routine to make sure the debugger does not modify the AIC context.

To summarize, in normal operating mode, the read of AIC_IVR performs the following operations within the AIC:

1. Calculates active interrupt (higher than current or spurious).
2. Determines and returns the vector of the active interrupt.
3. Memorizes the interrupt.
4. Pushes the current priority level onto the internal stack.
5. Acknowledges the interrupt.

However, while the Protect mode is activated, only operations 1 to 3 are performed when AIC_IVR is read. Operations 4 and 5 are only performed by the AIC when AIC_IVR is written.

Software that has been written and debugged using the Protect mode runs correctly in normal mode without modification. However, in normal mode, the AIC_IVR write has no effect and can be removed to optimize the code.

### 21.8.6 Spurious Interrupt

The AIC features a protection against spurious interrupts. A spurious interrupt is defined as being the assertion of an interrupt source long enough for the AIC to assert the nIRQ, but no longer present when AIC_IVR is read. This is most prone to occur when:

- An external interrupt source is programmed in Level-Sensitive mode and an active level occurs for only a short time.
- An internal interrupt source is programmed in level-sensitive and the output signal of the corresponding embedded peripheral is activated for a short time (as is the case for the watchdog).
- An interrupt occurs just a few cycles before the software begins to mask it, thus resulting in a pulse on the interrupt source.

The AIC detects a spurious interrupt at the time AIC_IVR is read while no enabled interrupt source is pending. When this happens, the AIC returns the value stored by the programmer in the Spurious Vector register (AIC_SPU). The programmer must store the address of a spurious interrupt handler in AIC_SPU as part of the application, to enable an as fast as possible return to the normal execution flow. This handler writes in AIC_EOICR and performs a return from interrupt.

### 21.8.7 General Interrupt Mask

The AIC features a General Interrupt Mask bit (AIC_DCR.GMSK) to prevent interrupts from reaching the processor. Both the nIRQ and the nFIQ lines are driven to their inactive state if AIC_DCR.GMSK is set. However, this mask does not prevent waking up the processor if it has entered Idle mode. This function facilitates synchronizing the processor on a next event and, as soon as the event occurs, performs subsequent operations without having to handle an interrupt. It is strongly recommended to use this mask with caution.

### 21.8.8 Register Write Protection

To prevent any single software error from corrupting AIC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the AIC Write Protection Mode Register (AIC_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the AIC Write Protection Status Register (AIC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading AIC_WPSR.

The following registers can be write-protected:

- AIC Source Mode Register
- AIC Source Vector Register
- AIC Spurious Interrupt Vector Register
- AIC Debug Control Register

## 21.9 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | AIC_SSR | 7:0 | | INTSEL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | AIC_SMR | 7:0 | | SRCTYPE[1:0] | | | | PRIOR[2:0] | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | AIC_SVR | 7:0 | VECTOR[7:0] | | | | | | | |
| | | 15:8 | VECTOR[15:8] | | | | | | | |
| | | 23:16 | VECTOR[23:16] | | | | | | | |
| | | 31:24 | VECTOR[31:24] | | | | | | | |
| 0x0C ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | AIC_IVR | 7:0 | IRQV[7:0] | | | | | | | |
| | | 15:8 | IRQV[15:8] | | | | | | | |
| | | 23:16 | IRQV[23:16] | | | | | | | |
| | | 31:24 | IRQV[31:24] | | | | | | | |
| 0x14 | AIC_FVR | 7:0 | FIQV[7:0] | | | | | | | |
| | | 15:8 | FIQV[15:8] | | | | | | | |
| | | 23:16 | FIQV[23:16] | | | | | | | |
| | | 31:24 | FIQV[31:24] | | | | | | | |
| 0x18 | AIC_ISR | 7:0 | | IRQID[6:0] | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C ... 0x1F | Reserved | | | | | | | | | |
| 0x20 | AIC_IPR0 | 7:0 | PID7 | PID6 | PID5 | PID4 | PID3 | PID2 | PID1 | FIQ |
| | | 15:8 | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| | | 23:16 | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| | | 31:24 | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| 0x24 | AIC_IPR1 | 7:0 | PID39 | PID38 | PID37 | PID36 | PID35 | PID34 | PID33 | PID32 |
| | | 15:8 | PID47 | PID46 | PID45 | PID44 | PID43 | PID42 | PID41 | PID40 |
| | | 23:16 | PID55 | PID54 | PID53 | PID52 | PID51 | PID50 | PID49 | PID48 |
| | | 31:24 | PID63 | PID62 | PID61 | PID60 | PID59 | PID58 | PID57 | PID56 |
| 0x28 | AIC_IPR2 | 7:0 | PID71 | PID70 | PID69 | PID68 | PID67 | PID66 | PID65 | PID64 |
| | | 15:8 | PID79 | PID78 | PID77 | PID76 | PID75 | SYS | PID73 | PID72 |
| | | 23:16 | PID87 | PID86 | PID85 | PID84 | PID83 | PID82 | PID81 | PID80 |
| | | 31:24 | PID95 | PID94 | PID93 | PID92 | PID91 | PID90 | PID89 | PID88 |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x2C | AIC_IPR3 | 7:0 | PID103 | PID102 | PID101 | PID100 | PID99 | PID98 | PID97 | PID96 |
| | | 15:8 | PID111 | PID110 | PID109 | PID108 | PID107 | PID106 | PID105 | PID104 |
| | | 23:16 | PID119 | PID118 | PID117 | PID116 | PID115 | PID114 | PID113 | PID112 |
| | | 31:24 | PID127 | PID126 | PID125 | PID124 | PID123 | PID122 | PID121 | PID120 |
| 0x30 | AIC_IMR | 7:0 | | | | | | | | INTM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x34 | AIC_CISR | 7:0 | | | | | | | NIRQ | NFIQ |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x38 | AIC_EOICR | 7:0 | | | | | | | | ENDIT |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x3C | AIC_SPU | 7:0 | SIVR[7:0] | | | | | | | |
| | | 15:8 | SIVR[15:8] | | | | | | | |
| | | 23:16 | SIVR[23:16] | | | | | | | |
| | | 31:24 | SIVR[31:24] | | | | | | | |
| 0x40 | AIC_IECR | 7:0 | | | | | | | | INTEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x44 | AIC_IDCR | 7:0 | | | | | | | | INTD |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x48 | AIC_ICCR | 7:0 | | | | | | | | INTCLR |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x4C | AIC_ISCR | 7:0 | | | | | | | | INTSET |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x50 ... 0x6B | Reserved | | | | | | | | | |
| 0x6C | AIC_DCR | 7:0 | | | | | | | GMSK | PROT |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x70<br>...<br>0xE3 | Reserved | | | | | | | | | |
| 0xE4 | AIC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | AIC_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |

### 21.9.1 AIC Source Select Register

**Name:** AIC_SSR
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | INTSEL[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 6:0 – INTSEL[6:0]** Interrupt Line Selection
0–127 = Selects the interrupt line to handle.

See the section ”Interrupt Source Mode”.

### 21.9.2 AIC Source Mode Register

**Name:** AIC_SMR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AIC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | SRCTYPE[1:0] | | | | PRIOR[2:0] | | |
| Access | | R/W | R/W | | | R/W | R/W | R/W |
| Reset | | 0 | 0 | | | 0 | 0 | 0 |

**Bits 6:5 – SRCTYPE[1:0]** Interrupt Source Type
The active level or edge is not programmable for the internal interrupt source selected by INTSEL.

| Value | Name | Description |
|---|---|---|
| 0 | INT_LEVEL_SENSITIVE | High-level sensitive for internal source.<br>Low-level sensitive for external source. |
| 1 | EXT_NEGATIVE_EDGE | Negative-edge triggered for external source. |
| 2 | EXT_HIGH_LEVEL | High-level sensitive for internal source.<br>High-level sensitive for external source. |
| 3 | EXT_POSITIVE_EDGE | Positive-edge triggered for external source. |

**Bits 2:0 – PRIOR[2:0]** Priority Level
Programs the priority level of the source selected by INTSEL except FIQ source (source 0).

The priority level can be between 0 (lowest) and 7 (highest).

The priority level is not used for the FIQ.

### 21.9.3 AIC Source Vector Register

**Name:** AIC_SVR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AIC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | VECTOR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | VECTOR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | VECTOR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | VECTOR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – VECTOR[31:0]** Source Vector
The user may store in this register the address of the corresponding handler for the interrupt source selected by INTSEL.

### 21.9.4 AIC Interrupt Vector Register

**Name:** AIC_IVR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IRQV[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IRQV[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IRQV[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IRQV[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – IRQV[31:0]** Interrupt Vector Register
The Interrupt Vector Register contains the vector programmed by the user in the Source Vector Register corresponding to the current interrupt.

The Source Vector Register is indexed using the current interrupt number when the Interrupt Vector Register is read.

When there is no current interrupt, the Interrupt Vector Register reads the value stored in AIC_SPU.

### 21.9.5 AIC FIQ Vector Register

**Name:** AIC_FVR
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FIQV[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FIQV[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FIQV[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FIQV[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – FIQV[31:0]** FIQ Vector Register
The FIQ Vector Register contains the vector programmed by the user in the Source Vector Register when INTSEL = 0. When there is no fast interrupt, the FIQ Vector Register reads the value stored in AIC_SPU.

### 21.9.6 AIC Interrupt Status Register

**Name:** AIC_ISR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | IRQID[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 6:0 – IRQID[6:0]** Current Interrupt Identifier
The Interrupt Status Register returns the current interrupt source number.

### 21.9.7 AIC Interrupt Pending Register 0

**Name:** AIC_IPR0
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read-only

The reset value of this register depends on the level of the external interrupt source. All other sources are cleared at reset, thus not pending.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID7 | PID6 | PID5 | PID4 | PID3 | PID2 | PID1 | FIQ |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32 – PIDx** Interrupt Pending
PID2...PID31 refer to the identifiers as defined in the Peripheral Identifiers section.

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is not pending. |
| 1 | The corresponding interrupt is pending. |

**Bit 0 – FIQ** Interrupt Pending

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is not pending. |
| 1 | The corresponding interrupt is pending. |

### 21.9.8 AIC Interrupt Pending Register 1

**Name:** AIC_IPR1
**Offset:** 0x24
**Reset:** 0x00000000
**Property:** Read-only

The reset value of this register depends on the level of the external interrupt source. All other sources are cleared at reset, thus not pending.
PID32...PID63 refer to the identifiers as defined in the Peripheral Identifiers section.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID63 | PID62 | PID61 | PID60 | PID59 | PID58 | PID57 | PID56 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID55 | PID54 | PID53 | PID52 | PID51 | PID50 | PID49 | PID48 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID47 | PID46 | PID45 | PID44 | PID43 | PID42 | PID41 | PID40 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID39 | PID38 | PID37 | PID36 | PID35 | PID34 | PID33 | PID32 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – PIDx** Interrupt Pending

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is not pending. |
| 1 | The corresponding interrupt is pending. |

### 21.9.9 AIC Interrupt Pending Register 2

**Name:** AIC_IPR2
**Offset:** 0x28
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID95 | PID94 | PID93 | PID92 | PID91 | PID90 | PID89 | PID88 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID87 | PID86 | PID85 | PID84 | PID83 | PID82 | PID81 | PID80 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID79 | PID78 | PID77 | PID76 | PID75 | SYS | PID73 | PID72 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID71 | PID70 | PID69 | PID68 | PID67 | PID66 | PID65 | PID64 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – PIDx** Interrupt Pending

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is not pending. |
| 1 | The corresponding interrupt is pending. |

**Bit 10 – SYS** Interrupt Pending

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is not pending. |
| 1 | The corresponding interrupt is pending. |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 – PIDx** Interrupt Pending

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is not pending. |
| 1 | The corresponding interrupt is pending. |

### 21.9.10 AIC Interrupt Pending Register 3

**Name:** AIC_IPR3
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** Read-only

The reset value of this register depends on the level of the external interrupt source. All other sources are cleared at reset, thus not pending.
PID96...PID127 bit fields refer to the identifiers as defined in the Peripheral Identifiers section.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID127 | PID126 | PID125 | PID124 | PID123 | PID122 | PID121 | PID120 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID119 | PID118 | PID117 | PID116 | PID115 | PID114 | PID113 | PID112 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID111 | PID110 | PID109 | PID108 | PID107 | PID106 | PID105 | PID104 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID103 | PID102 | PID101 | PID100 | PID99 | PID98 | PID97 | PID96 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – PIDx** Interrupt Pending

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is not pending. |
| 1 | The corresponding interrupt is pending. |

### 21.9.11 AIC Interrupt Mask Register

**Name:** AIC_IMR
**Offset:** 0x30
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | INTM |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 0 – INTM** Interrupt Mask

| Value | Description |
|---|---|
| 0 | The interrupt source selected by AIC_SSR.INTSEL is disabled. |
| 1 | The interrupt source selected by AIC_SSR.INTSEL is enabled. |

### 21.9.12 AIC Core Interrupt Status Register

**Name:** AIC_CISR
**Offset:** 0x34
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | NIRQ | NFIQ |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – NIRQ** NIRQ Status

| Value | Description |
|---|---|
| 0 | nIRQ line is deactivated. |
| 1 | nIRQ line is active. |

**Bit 0 – NFIQ** NFIQ Status

| Value | Description |
|---|---|
| 0 | nFIQ line is deactivated. |
| 1 | nFIQ line is active. |

### 21.9.13 AIC End of Interrupt Command Register

**Name:** AIC_EOICR
**Offset:** 0x38
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ENDIT |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – ENDIT** Interrupt Processing Complete Command
The End of Interrupt Command Register is used by the interrupt routine to indicate that the interrupt treatment is complete. Any value can be written because it is only necessary to make a write to this register location to signal the end of interrupt treatment.

### 21.9.14 AIC Spurious Interrupt Vector Register

**Name:** AIC_SPU
**Offset:** 0x3C
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AIC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SIVR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SIVR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SIVR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SIVR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – SIVR[31:0]** Spurious Interrupt Vector Register
The user may store the address of a spurious interrupt handler in this register. The written value is returned in AIC_IVR in case of a spurious interrupt, or in AIC_FVR in case of a spurious fast interrupt.

### 21.9.15 AIC Interrupt Enable Command Register

**Name:** AIC_IECR
**Offset:** 0x40
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | INTEN |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – INTEN** Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the interrupt source selected by AIC_SSR.INTSEL. |

### 21.9.16 AIC Interrupt Disable Command Register

**Name:** AIC_IDCR
**Offset:** 0x44
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | INTD |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – INTD** Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the interrupt source selected by AIC_SSR.INTSEL. |

### 21.9.17 AIC Interrupt Clear Command Register

**Name:** AIC_ICCR
**Offset:** 0x48
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | INTCLR |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – INTCLR** Interrupt Clear
Clears one the following depending on the setting of AIC_SSR.INTSEL: FIQ, SYS, PID2-PID73 and PID75-PID127

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears the interrupt source selected by AIC_SSR.INTSEL. |

### 21.9.18 AIC Interrupt Set Command Register

**Name:** AIC_ISCR
**Offset:** 0x4C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | INTSET |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – INTSET** Interrupt Set

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Sets the interrupt source selected by INTSEL. |

### 21.9.19 AIC Debug Control Register

**Name:** AIC_DCR
**Offset:** 0x6C
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AIC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | GMSK | PROT |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – GMSK** General Interrupt Mask

| Value | Description |
|---|---|
| 0 | The nIRQ and nFIQ lines are normally controlled by the AIC. |
| 1 | The nIRQ and nFIQ lines are tied to their inactive state. |

**Bit 0 – PROT** Protection Mode

| Value | Description |
|---|---|
| 0 | The Protection mode is disabled. |
| 1 | The Protection mode is enabled. |

### 21.9.20 AIC Write Protection Mode Register

**Name:** AIC_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x414943 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit.<br>Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See section ”Register Write Protection” for the list of registers that can be protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x414943 (“AIC” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x414943 (“AIC” in ASCII). |

### 21.9.21 AIC Write Protection Status Register

**Name:** AIC_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of AIC_WPSR. |
| 1 | A write protection violation has occurred since the last read of AIC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 22. Watchdog Timer (WDT)

## 22.1 Description

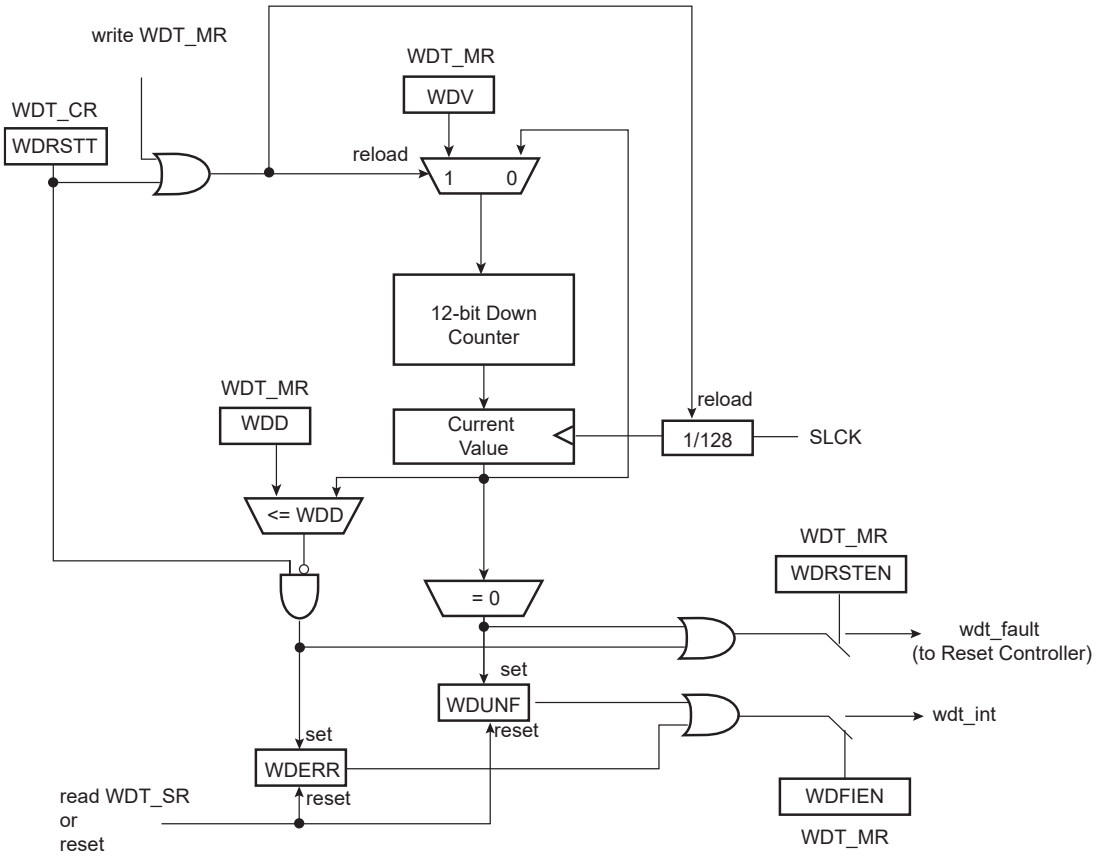
The Watchdog Timer (WDT) is used to prevent system lock-up if the software becomes trapped in a deadlock. It features a 12-bit down counter that allows a watchdog period of up to 16 seconds (slow clock around 32 kHz). It can generate a general reset or a processor reset only. In addition, it can be stopped while the processor is in Debug mode or Sleep mode (Idle mode).

## 22.2 Embedded Characteristics

- 12-bit Key-protected Programmable Counter
- Watchdog Clock is Independent from Processor Clock
- Provides Reset or Interrupt Signals to the System
- Counter May Be Stopped while the Processor is in Debug State or in Idle Mode

## 22.3 Block Diagram

**Figure 22-1. Watchdog Timer Block Diagram**



## 22.4 Functional Description

The Watchdog Timer is used to prevent system lock-up if the software becomes trapped in a deadlock. It is supplied with VDDCORE. It restarts with initial values on processor reset.

The watchdog is built around a 12-bit down counter, which is loaded with the value defined in the field WDV of the Mode Register (WDT_MR). The Watchdog Timer uses the slow clock divided by 128 to establish the maximum watchdog period to be 16 seconds (with a typical slow clock of 32.768 kHz).

After a processor reset, the value of WDV is 0xFFF, corresponding to the maximum value of the counter with the external reset generation enabled (field WDRSTEN at 1 after a backup reset). This means that a default watchdog is running at reset, i.e., at power-up. The user can either disable the WDT by setting bit WDT_MR.WDDIS or reprogram the WDT to meet the maximum watchdog period the application requires.

When setting the WDDIS bit, and while it is set, the fields WDV and WDD must not be modified.

If the watchdog is restarted by writing into the Control Register (WDT_CR), WDT_MR must not be programmed during a period of time of three slow clock periods following the WDT_CR write access. In any case, programming a new value in WDT_MR automatically initiates a restart instruction.

WDT_MR can be written until a LOCKMR command is issued in WDT_CR. Only a processor reset resets it. Writing WDT_MR reloads the timer with the newly programmed mode parameters.

In normal operation, the user reloads the watchdog at regular intervals before the timer underflow occurs, by setting bit WDT_CR.WDRSTT. The watchdog counter is then immediately reloaded from WDT_MR and restarted, and the slow clock 128 divider is reset and restarted. WDT_CR is write-protected. As a result, writing WDT_CR without the correct hard-coded key has no effect. If an underflow does occur, the “wdt_fault” signal to the Reset Controller is asserted if bit WDT_MR.WDRSTEN is set. Moreover, the bit WDUNF is set in the Status Register (WDT_SR).

The reload of the watchdog must occur while the watchdog counter is within a window between 0 and WDD. WDD is defined in WDT_MR.

Any attempt to restart the watchdog while the watchdog counter is between WDV and WDD results in a watchdog error, even if the watchdog is disabled. The bit WDT_SR.WDERR is updated and the “wdt_fault” signal to the Reset Controller is asserted.

Note that this feature can be disabled by programming a WDD value greater than or equal to the WDV value. In such a configuration, restarting the Watchdog Timer is permitted in the whole range [0; WDV] and does not generate an error. This is the default configuration on reset (the WDD and WDV values are equal).

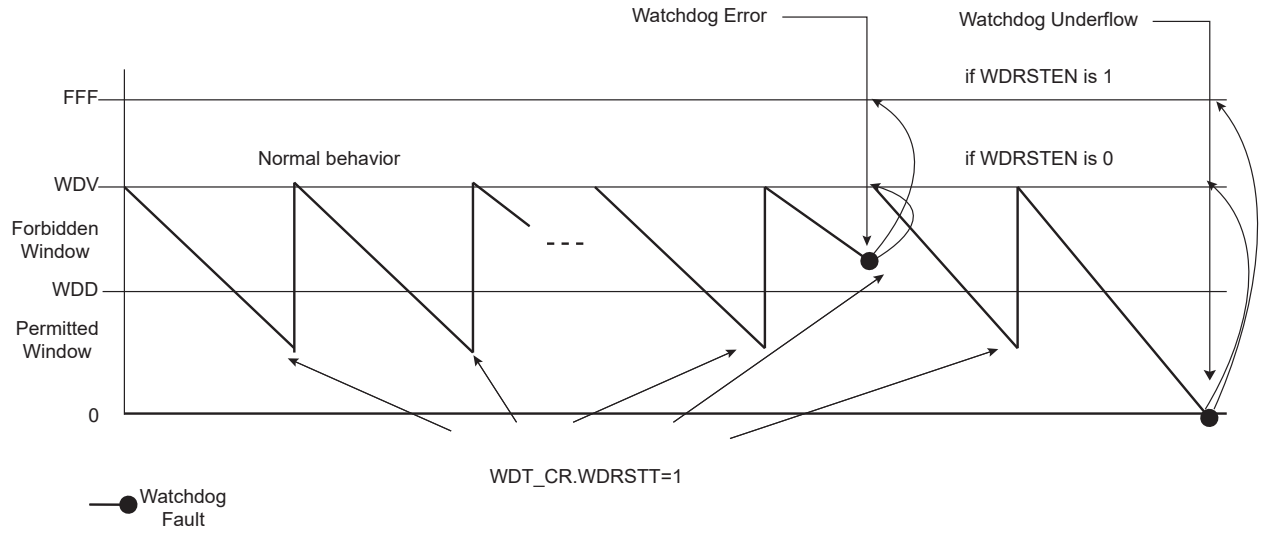
The status bits WDUNF (Watchdog Underflow) and WDERR (Watchdog Error) trigger an interrupt, provided the bit WDT_MR.WDFIEN is set. The signal “wdt_fault” to the Reset Controller causes a watchdog reset if the WDRSTEN bit is set as already explained in the Reset Controller documentation. In this case, the processor and the Watchdog Timer are reset, and the WDERR and WDUNF flags are reset.

If a reset is generated or if WDT_SR is read, the status bits are reset, the interrupt is cleared, and the “wdt_fault” signal to the reset controller is deasserted.

Writing WDT_MR reloads and restarts the down counter.

While the processor is in debug state or in Sleep mode, the counter may be stopped depending on the value programmed for the bits WDIDLEHLT and WDDBGHLT in WDT_MR.

**Figure 22-2. Watchdog Behavior**



## 22.5 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | WDT_CR | 7:0 | | | | LOCKMR | | | | WDRSTT |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | KEY[7:0] | | | | | | | |
| 0x04 | WDT_MR | 7:0 | WDV[7:0] | | | | | | | |
| | | 15:8 | WDDIS | | WDRSTEN | WDFIEN | WDV[11:8] | | | |
| | | 23:16 | WDD[7:0] | | | | | | | |
| | | 31:24 | | | WDIDLEHLT | WDDBGHLT | WDD[11:8] | | | |
| 0x08 | WDT_SR | 7:0 | | | | | | | WDERR | WDUNF |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 22.5.1 Watchdog Timer Control Register

**Name:** WDT_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

The WDT_CR register values must not be modified within three slow clock periods following a restart of the watchdog performed by a write access in WDT_CR. Any modification will cause the watchdog to trigger an end of period earlier than expected.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | LOCKMR | | | | WDRSTT |
| Access | | | | W | | | | W |
| Reset | | | | – | | | | – |

**Bits 31:24 – KEY[7:0]** Password

| Value | Name | Description |
|---|---|---|
| 0xA5 | PASSWD | Writing any other value in this field aborts the write operation. |

**Bit 4 – LOCKMR** Lock Mode Register Write Access

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Locks the Mode Register (WDT_MR) if KEY is written to 0xA5, write access to WDT_MR has no effect. |

**Bit 0 – WDRSTT** Watchdog Restart

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Restarts the watchdog if KEY is written to 0xA5. |

### 22.5.2 Watchdog Timer Mode Register

**Name:** WDT_MR
**Offset:** 0x04
**Reset:** 0x3FFF2FFF
**Property:** Read/Write

Write access to this register has no effect if the LOCKMR command is issued in WDT_CR (unlocked on hardware reset).

The WDT_MR register values must not be modified within three slow clock periods following a restart of the watchdog performed by a write access in WDT_CR. Any modification will cause the watchdog to trigger an end of period earlier than expected.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | WDIDLEHLT | WDDBGHLT | WDD[11:8] | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WDD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WDDIS | | WDRSTEN | WDFIEN | WDV[11:8] | | | |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 1 | 0 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WDV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bit 29 – WDIDLEHLT** Watchdog Idle Halt

| Value | Description |
|---|---|
| 0 | The watchdog runs when the system is in idle state. |
| 1 | The watchdog stops when the system is in idle state. |

**Bit 28 – WDDBGHLT** Watchdog Debug Halt

| Value | Description |
|---|---|
| 0 | The watchdog runs when the processor is in debug state. |
| 1 | The watchdog stops when the processor is in debug state. |

**Bits 27:16 – WDD[11:0]** Watchdog Delta Value
Defines the permitted range for reloading the Watchdog Timer.

If the Watchdog Timer value is less than or equal to WDD, setting bit WDT_CR.WDRSTT restarts the timer.

If the Watchdog Timer value is greater than WDD, setting bit WDT_CR.WDRSTT causes a watchdog error.

**Bit 15 – WDDIS** Watchdog Disable
When setting the WDDIS bit, and while it is set, the fields WDV and WDD must not be modified.

| Value | Description |
|---|---|
| 0 | Enables the Watchdog Timer. |
| 1 | Disables the Watchdog Timer. |

**Bit 13 – WDRSTEN** Watchdog Reset Enable

| Value | Description |
|---|---|
| 0 | A watchdog fault (underflow or error) has no effect on the resets. |
| 1 | A watchdog fault (underflow or error) triggers a watchdog reset. |

**Bit 12 – WDFIEN** Watchdog Fault Interrupt Enable

| Value | Description |
|---|---|
| 0 | A watchdog fault (underflow or error) has no effect on interrupt. |
| 1 | A watchdog fault (underflow or error) asserts interrupt. |

**Bits 11:0 – WDV[11:0]** Watchdog Counter Value
Defines the value loaded in the 12-bit watchdog counter.

### 22.5.3 Watchdog Timer Status Register

**Name:** WDT_SR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | WDERR | WDUNF |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – WDERR** Watchdog Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No watchdog error occurred since the last read of WDT_SR. |
| 1 | At least one watchdog error occurred since the last read of WDT_SR. |

**Bit 0 – WDUNF** Watchdog Underflow (cleared on read)

| Value | Description |
|---|---|
| 0 | No watchdog underflow occurred since the last read of WDT_SR. |
| 1 | At least one watchdog underflow occurred since the last read of WDT_SR. |

# 23. Reset Controller (RSTC)

## 23.1 Description

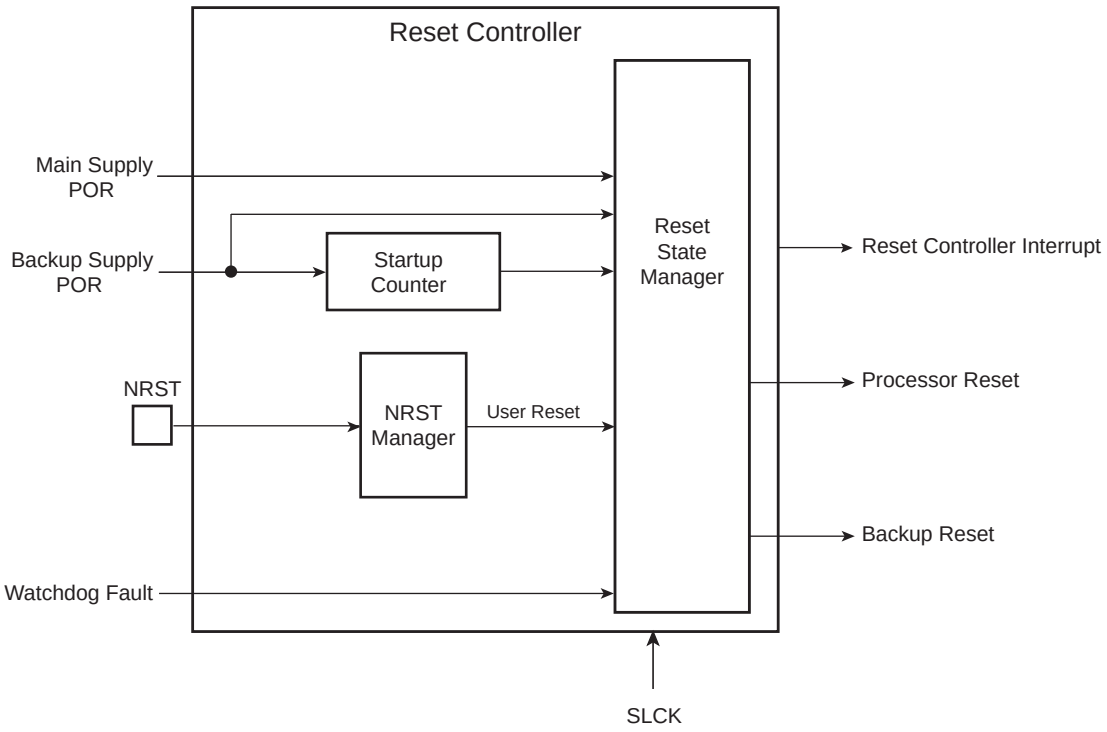
The Reset Controller (RSTC), based on power-on reset cells, handles all the resets of the system without any external components. It reports which reset occurred last.

## 23.2 Embedded Characteristics

- Manages All Resets of the System, Including
  - Processor reset
  - Backed-up peripheral reset
- Based on Two Embedded Power-on Reset Cells
- Reset Source Status
  - Status of the last reset
  - Either general reset, wake-up reset, software reset, user reset, watchdog reset

## 23.3 Block Diagram

**Figure 23-1. Reset Controller Block Diagram**



## 23.4 Functional Description

### 23.4.1 Reset Controller Overview

The Reset Controller (RSTC) is made up of an NRST manager, a startup counter and a reset state manager. It runs at Slow Clock and generates the following reset signals:

- Processor reset–Resets the processor and the entire set of embedded peripherals.
- Backup reset–Resets all the peripherals powered by VDDBU.

These reset signals are asserted by the RSTC, either on external events or on software action. The reset state manager controls the generation of reset signals.

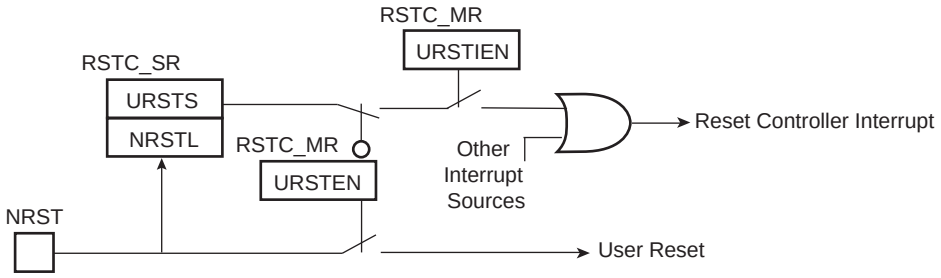
The startup counter waits for the complete crystal oscillator startup. The wait delay is given by the crystal oscillator startup time maximum value that can be found in the section “Crystal Oscillator Characteristics” in “Electrical Characteristics”.

The Mode register (RSTC_MR), used to configure the reset controller, is powered with VDDBU, so that its configuration is saved as long as VDDBU is on.

### 23.4.2 NRST Manager

The NRST manager samples the NRST input pin and drives this pin low when required by the reset state manager. The figure below shows the block diagram of the NRST manager.

**Figure 23-2. NRST Manager**




#### 23.4.2.1 NRST Signal or Interrupt

The NRST manager samples the NRST pin at Slow Clock speed. When the line is detected low, a user reset is reported to the reset state manager.

However, the NRST manager can be programmed to not trigger a reset when an assertion of NRST occurs. Writing a zero to RSTC_MR.URSTEN disables the user reset trigger.

The level of the pin NRST can be read at any time in the bit NRSTL (NRST level) in the Status register (RSTC_SR). As soon as the pin NRST is asserted, RSTC_SR.URSTS is set. This bit clears only when RSTC_SR is read.

The RSTC can also be programmed to generate an interrupt instead of generating a reset. To do so, RSTC_MR.URSTIEN must be set.

### 23.4.3 Reset States

The reset state manager handles the different reset sources and generates the internal reset signals. It reports the reset status in RSTC_SR.RSTTYP. The update of RSTTYP is performed when the processor reset is released.

#### 23.4.3.1 General Reset

A general reset occurs when VDDBU and VDDCORE are powered on. The backup supply POR cell output rises and is filtered with a startup counter, which operates at Slow Clock. The purpose of this counter is to make sure the Slow Clock oscillator is stable before starting up the device. The length of startup time is hardcoded to comply with the Slow Clock Oscillator startup time.

After this time, the processor clock is released at Slow Clock and all the other signals remain valid for two cycles for proper processor and logic reset. Then, all the reset signals are released and RSTC_SR.RSTTYP reports a general reset.

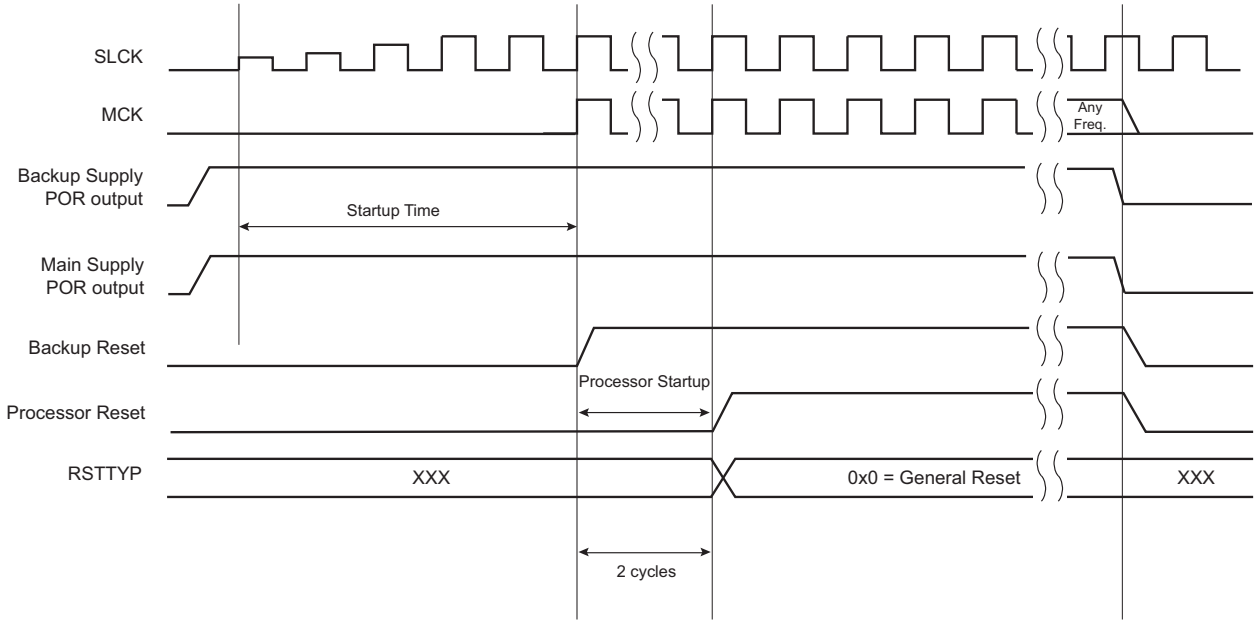
When VDDBU is detected low by the backup supply POR cell, all resets signals are immediately asserted, even if the main supply POR cell does not report a main supply shutdown.

VDDBU only activates the backup reset signal.

Backup reset must be released so that any other reset can be generated by VDDCORE (main supply POR output).

The figure below shows how the general reset affects the reset signals.

**Figure 23-3. General Reset State**



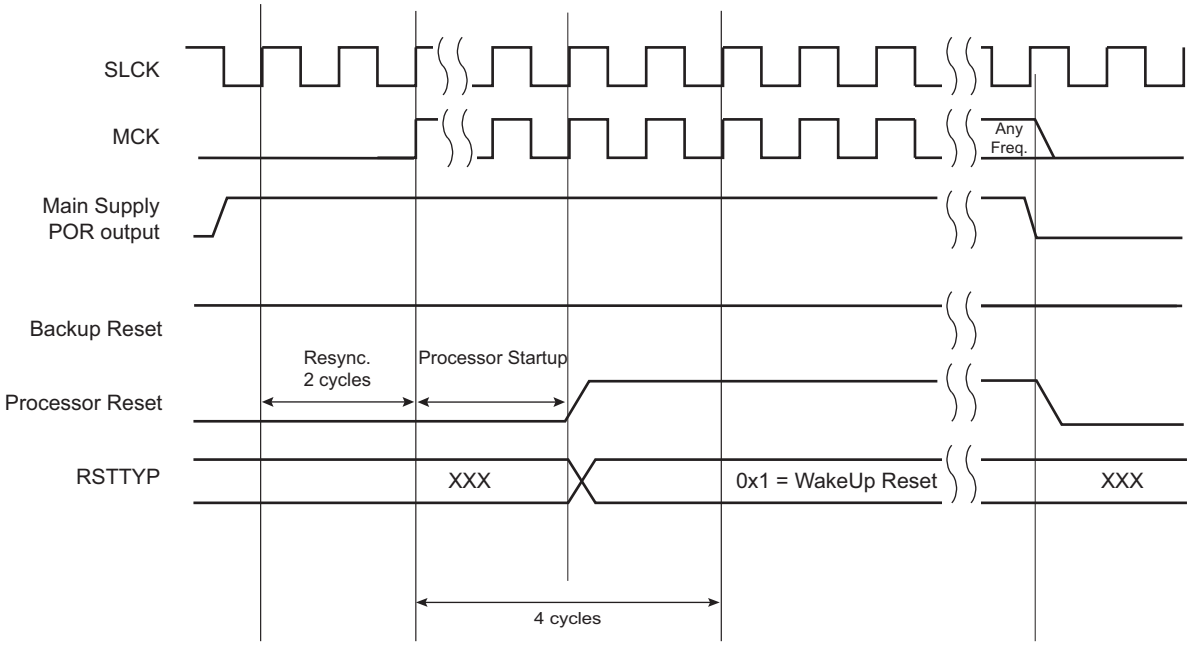

#### 23.4.3.2 Wake-up Reset

The wake-up reset occurs when the main supply is down. When the main supply POR output is active, all the reset signals are asserted except backup reset. When the main supply powers up, the POR output is resynchronized on Slow Clock. The processor clock is then re-enabled during two Slow Clock cycles, depending on the requirements of the ARM processor.

At the end of this delay, the processor and other reset signals rise. RSTC_SR.RSTTYP is updated to report a wake-up reset.

When the main supply is detected falling, the reset signals are immediately asserted. This transition is synchronous with the output of the main supply POR.

**Figure 23-4. Wake-up Reset**




### 23.4.3.3 User Reset

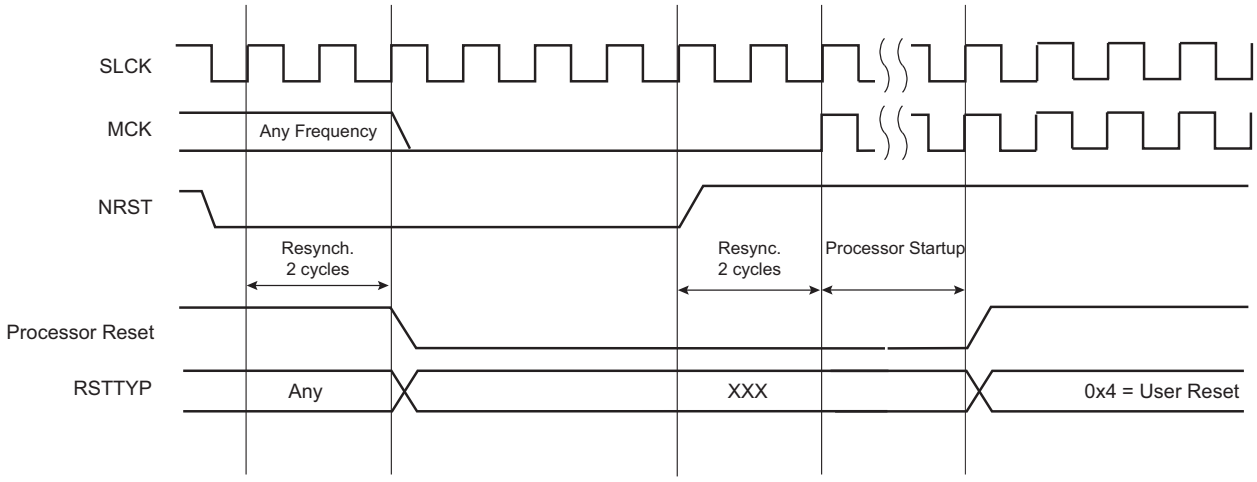
The user reset is entered when a low level is detected on the NRST pin and RSTC_MR.URSTEN is at 1. The NRST input signal is resynchronized with SLCK to ensure proper behavior of the system.

The processor reset and the peripheral reset are asserted.

The user reset is left when NRST rises, after a two-cycle resynchronization time and a two-cycle processor startup. The processor clock is re-enabled as soon as NRST is confirmed high.

When the processor reset signal is released, RSTC_SR.RSTTYP is loaded with the value 0x4, indicating a user reset.

**Figure 23-5. User Reset State**




### 23.4.3.4 Software Reset

The RSTC offers several commands used to assert the different reset signals. These commands are performed by writing the Control register (RSTC_CR) with the following bits at 1:

- PROCRST–Writing PROCRST at 1 resets the processor, the watchdog timer and all the embedded peripherals, including the memory system, and, in particular, the remap command.

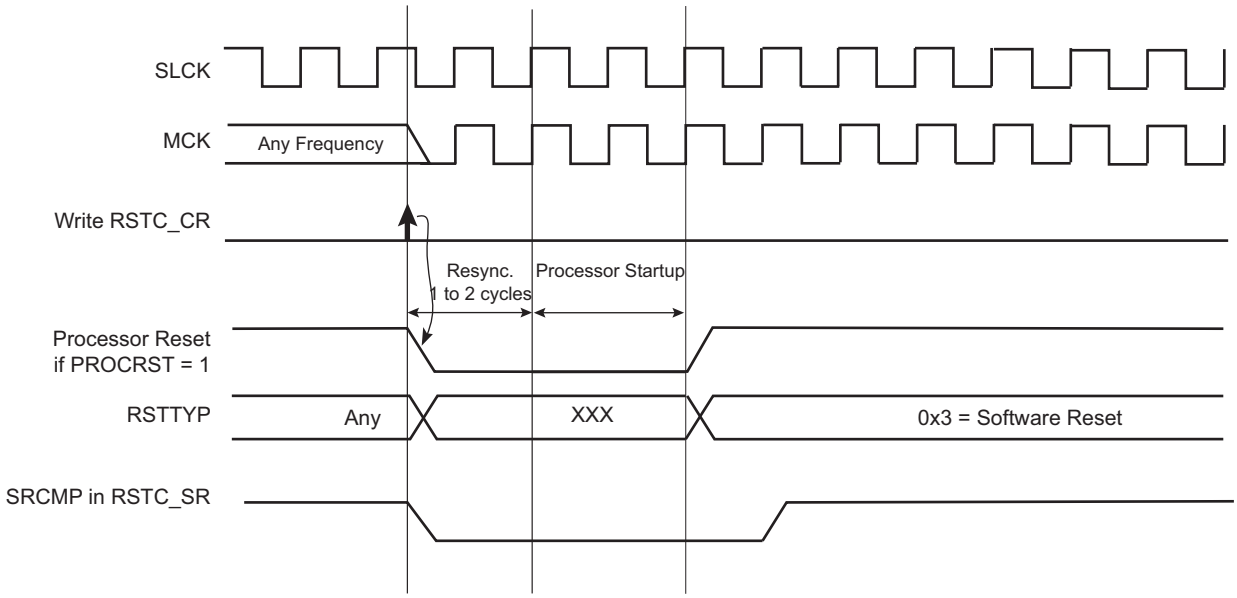
The software reset is entered if at least one of these bits is set by the software. All these commands can be performed independently or simultaneously. The software reset lasts two Slow Clock cycles.

The internal reset signals are asserted as soon as the register write is performed. This is detected on the Master Clock (MCK). They are released when the software reset is left, i.e., synchronously to SLCK.

If and only if RSTC_CR.PROCRST is set, the RSTC reports the software status in RSTC_SR.RSTTYP. Other software resets are not reported in RSTTYP.

As soon as a software operation is detected, RSTC_SR.SRCMP is set. It is cleared as soon as the software reset is left. No other software reset can be performed while RSTC_SR.SRCMP is set, and writing any value in RSTC_CR has no effect.

**Figure 23-6. Software Reset**



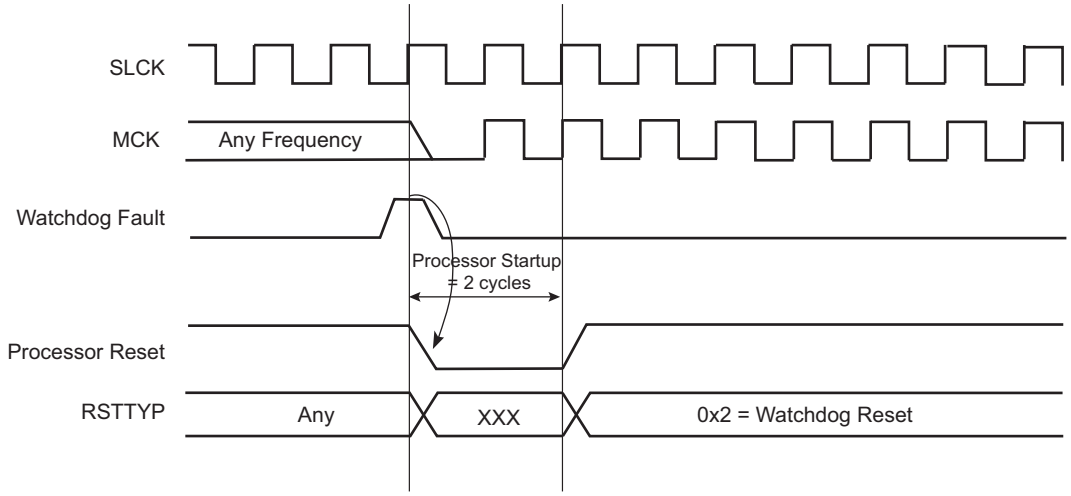

#### 23.4.3.5 Watchdog Reset

The watchdog reset is entered when a watchdog fault occurs. This state lasts two Slow Clock cycles.

The watchdog timer is reset by the processor reset signal. As the watchdog fault always causes a processor reset if WDT_MR.WDRSTEN is set, the watchdog timer is always reset after a watchdog reset and the watchdog is enabled by default and with a period set to a maximum.

When WDT_MR.WDRSTEN is reset, the watchdog fault has no impact on the reset controller.

**Figure 23-7. Watchdog Reset**




### 23.4.4 Reset State Priorities

The reset state manager manages the following priorities between the different reset sources, given in descending order:

- Backup reset
- Wake-up reset
- Watchdog reset
- Software reset
- User reset

Particular cases are listed below:

- When in user reset:
  - A watchdog event is impossible because the watchdog timer is being reset by the processor reset signal.
  - A software reset is impossible, since the processor reset is being activated.
- When in software reset:
  - A watchdog event has priority over the current state.
  - The NRST has no effect.
- When in watchdog reset:
  - The processor reset is active and so a software reset cannot be programmed.
  - A user reset cannot be entered.

## 23.5 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | RSTC_CR | 7:0 | | | | | | | | PROCRST |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | KEY[7:0] | | | | | | | |
| 0x04 | RSTC_SR | 7:0 | | | | | | | | URSTS |
| | | 15:8 | | | | | | RSTTYP[2:0] | | |
| | | 23:16 | | | | | | | SRCMP | NRSTL |
| | | 31:24 | | | | | | | | |
| 0x08 | RSTC_MR | 7:0 | | | | URSTIEN | | | | URSTEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | KEY[7:0] | | | | | | | |

### 23.5.1 Reset Controller Control Register

**Name:** RSTC_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | PROCRST |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bits 31:24 – KEY[7:0]** Write Access Password

| Value | Name | Description |
|---|---|---|
| 0xA5 | PASSWD | Writing any other value in this field aborts the write operation.<br>Always reads as 0. |

**Bit 0 – PROCRST** Processor Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If KEY value = 0xA5, resets the processor and the peripherals<br>. |

### 23.5.2 Reset Controller Status Register

**Name:** RSTC_SR
**Offset:** 0x04
**Reset:** See Note
**Property:** Read-only

**Note:** Register reset values are:
0x00000100 only when VDDCORE is rising

0x00000000 when both power supplies VDDCORE and VDDBU are rising (backup reset)

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | R | R |
| Reset | | | | | | | SRCMP | NRSTL |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | R | R | R |
| Reset | | | | | | RSTTYP[2:0] | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | R |
| Reset | | | | | | | | URSTS |

**Bit 17 – SRCMP** Software Reset Command in Progress

| Value | Description |
|---|---|
| 0 | No software command is being performed by the RSTC. The RSTC is ready for a software command. |
| 1 | A software reset command is being performed by the RSTC. The RSTC is busy. |

**Bit 16 – NRSTL** NRST Pin Level
Records the level of the NRST pin sampled on each Master Clock (MCK) rising edge.

**Bits 10:8 – RSTTYP[2:0]** Reset Type
Reports the cause of the last processor reset. Reading RSTC_SR does not reset this field.

| Value | Name | Description |
|---|---|---|
| 0 | GENERAL_RST | Both VDDCORE and VDDBU rising |
| 1 | WKUP_RST | VDDCORE rising |
| 2 | WDT_RST | Watchdog fault occurred |
| 3 | SOFT_RST | Processor reset required by the software |

| Value | Name | Description |
|---|---|---|
| 4 | USER_RST | NRST pin detected low |

**Bit 0 – URSTS** User Reset Status

| Value | Description |
|---|---|
| 0 | No high-to-low edge on NRST happened since the last read of RSTC_SR. |
| 1 | At least one high-to-low transition of NRST has been detected since the last read of RSTC_SR. Reading the RSTC_SR resets URSTS and clears the interrupt. |

### 23.5.3 Reset Controller Mode Register

**Name:** RSTC_MR
**Offset:** 0x08
**Reset:** See Note
**Property:** Read/Write

**Note:**
Backup reset value is 0x00000000

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | URSTIEN | | | | URSTEN |
| Access | | | | R/W | | | | R/W |
| Reset | | | | | | | | |

**Bits 31:24 – KEY[7:0]** Write Access Password

| Value | Name | Description |
|---|---|---|
| 0xA5 | PASSWD | Writing any other value in this field aborts the write operation.<br>Always reads as 0. |

**Bit 4 – URSTIEN** User Reset Interrupt Enable

| Value | Description |
|---|---|
| 0 | RSTC_SR.URSTS at 1 has no effect on the RSTC interrupt. |
| 1 | RSTC_SR.URSTS at 1 asserts the RSTC interrupt if URSTEN = 0. |

**Bit 0 – URSTEN** User Reset Enable

| Value | Description |
|---|---|
| 0 | The detection of a low level on the pin NRST does not trigger a user reset. |
| 1 | The detection of a low level on the pin NRST triggers a user reset. |

# 24. Shutdown Controller (SHDWC)

## 24.1 Description

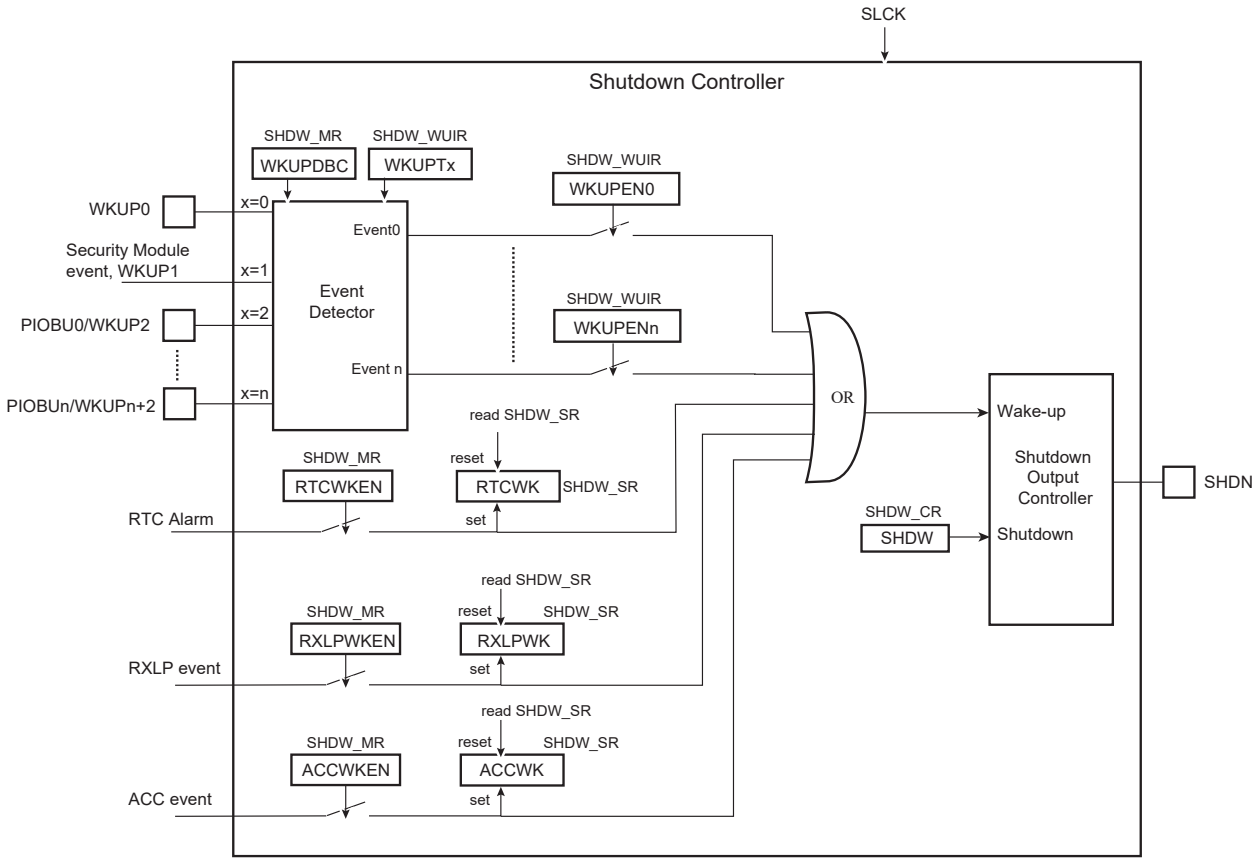
The Shutdown Controller (SHDWC) controls the VDDIO and VDDCORE power supplies and the wakeup detection on debounced input lines.

## 24.2 Embedded Characteristics

- Shutdown Logic
  - Software Assertion of the Shutdown Output Pin (SHDN)
  - Programmable De-assertion from the PIOBU, WKUP Input Pins
- Wakeup Logic
  - Programmable Assertion from the PIOBU, WKUP Input Pins, and Internal Wakeup Event from RTC, RXLP, ACC, Security Module

## 24.3 Block Diagram

**Figure 24-1. Shutdown Controller Block Diagram**



## 24.4 I/O Lines Description

**Table 24-1. I/O Lines Description**

| Name | Description | Type |
|---|---|---|
| WKUP0 | Wakeup inputs | Input |
| PIOBU 0-7 | Wakeup inputs, WKUP(2-9) | Input |
| SHDN | Shutdown output | Output |

## 24.5 Product Dependencies

### 24.5.1 Power Management

The Shutdown Controller is continuously clocked by the Slow Clock (SLCK). The Power Management Controller has no effect on the behavior of the Shutdown Controller.

## 24.6 Functional Description

The Shutdown Controller manages the main power supply. To do so, it is supplied with VDDBU and manages wakeup input pins and one output pin, SHDN.

A typical application connects the pin SHDN to the shutdown input of the DC/DC Converter providing the main power supplies of the system, and especially VDDCORE and/or VDDIO. The wakeup inputs (WKUPn) connect to any push-buttons or signal that wake up the system.

The software is able to control the pin SHDN by writing the Shutdown Control Register (SHDW_CR) with the bit SHDW at 1. The shutdown is taken into account only two slow clock cycles after the write of SHDW_CR. This register is password-protected and so the value written should contain the correct key for the command to be taken into account. As a result, the system should be powered down.

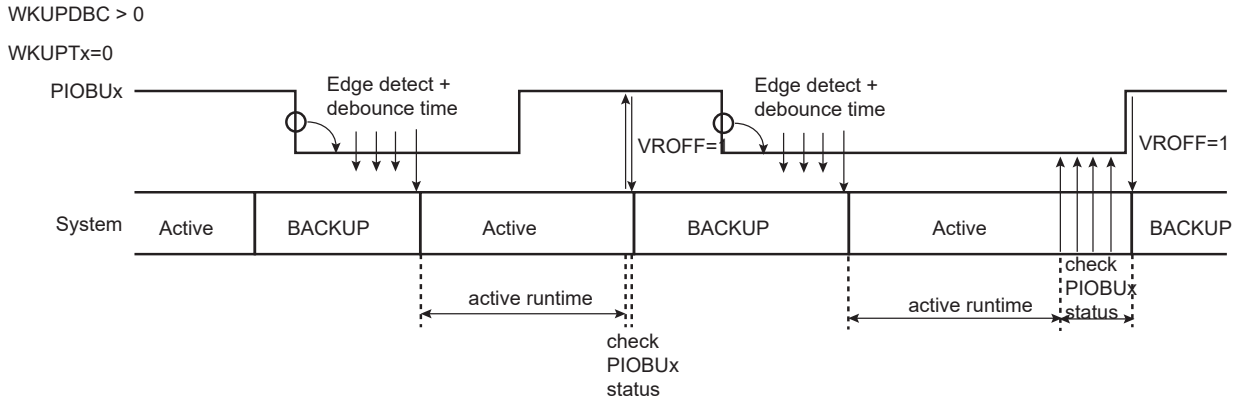
### 24.6.1 Wakeup Inputs

Any level change on a PIOBUx, WKUP pin, or Security Module event, can trigger a wakeup. Wakeup is configured in the Shutdown Mode Register (SHDW_MR) and Shutdown Wakeup Inputs Register (SHDW_WUIR). The transition detector can be programmed to detect either a positive or negative transition on any PIOBUx, WKUP pin. The detection can also be disabled. Programming is performed by enabling the Wakeup Input (WKUPENx bit) and defining the Wakeup Input Type (WKUPTx bit) in the SHDW_WUIR.

Moreover, a debouncing circuit can be programmed for PIOBUx, WKUP. The debouncing circuit filters pulses on PIOBUx, WKUP shorter than the programmed value in the WKUPDBC field in SHDW_MR. If the programmed level change is detected on a pin, a counter starts. When the counter reaches the value programmed in the corresponding field WKUPDBC, the SHDN pin is released. If a new input change is detected before the counter reaches the corresponding value, the counter is stopped and cleared. One counter is shared among all PIOBUx, WKUP inputs and all programmed level detection is merged into this counter. The WKUPISx bit of the Status Register (SHDW_SR) reports the detection of the programmed events on PIOBUx, WKUP with a reset after the read of SHDW_SR.

© 2018 Microchip Technology Inc.

**Figure 24-2. Entering and Exiting Backup Mode with a PIOBUx, WKUP Pin**




The Shutdown Controller can be programmed so as to activate the wakeup using the RTC alarm, RXLP event, ACC comparison event, security module event (detection of the rising edge event is synchronized with SLCK). This is done by writing the SHDW_MR using the RTCWKEN bit, RXLPWKEN bit and ACCWKEN bit. When enabled, the detection of RTC alarm, RXLP event, ACC comparison event, security module event is reported in the RTCWK bit, RXLPWK and ACCWK bits of SHDW_SR. They are cleared after reading SHDW_SR. When using the RTC alarm to wake up the system, the user must ensure that RTC alarm, RXLPWK and ACCWK status flags are cleared before shutting down the system. Otherwise, no rising edge of the status flags may be detected and the wakeup will fail.

## 24.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | SHDW_CR | 7:0 | | | | | | | | SHDW |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | KEY[7:0] | | | | | | | |
| 0x04 | SHDW_MR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | RXLPWKEN | ACCWKEN | RTCWKEN | |
| | | 31:24 | | | | | | WKUPDBC[2:0] | | |
| 0x08 | SHDW_SR | 7:0 | RXLPWK | ACCWK | RTCWK | | | | | WKUPS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | WKUPIS7 | WKUPIS6 | WKUPIS5 | WKUPIS4 | WKUPIS3 | WKUPIS2 | WKUPIS1 | WKUPIS0 |
| | | 31:24 | | | | | | | WKUPIS9 | WKUPIS8 |
| 0x0C | SHDW_WUIR | 7:0 | WKUPEN7 | WKUPEN6 | WKUPEN5 | WKUPEN4 | WKUPEN3 | WKUPEN2 | WKUPEN1 | WKUPEN0 |
| | | 15:8 | | | | | | | WKUPEN9 | WKUPEN8 |
| | | 23:16 | WKUPT7 | WKUPT6 | WKUPT5 | WKUPT4 | WKUPT3 | WKUPT2 | WKUPT1 | WKUPT0 |
| | | 31:24 | | | | | | | WKUPT9 | WKUPT8 |

### 24.7.1 Shutdown Controller Control Register

**Name:** SHDW_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SHDW |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bits 31:24 – KEY[7:0]** Password

| Value | Name | Description |
|---|---|---|
| 0xA5 | PASSWD | Writing any other value in this field aborts the write operation. |

**Bit 0 – SHDW** Shutdown Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If KEY value is correct, asserts the SHDN pin. |

### 24.7.2 Shutdown Controller Mode Register

**Name:** SHDW_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | WKUPDBC[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RXLPWKEN | ACCWKEN | RTCWKEN | |
| Access | | | | | W | W | R/W | |
| Reset | | | | | 0 | 0 | 0 | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 26:24 – WKUPDBC[2:0]** Wakeup Inputs Debouncer Period

| Value | Name | Description |
|---|---|---|
| 0 | IMMEDIATE | Immediate, no debouncing, detected active at least on one Slow Clock edge |
| 1 | 3_SLCK | PIOBUx shall be in its active state for at least 3 SLCK periods |
| 2 | 32_SLCK | PIOBUx shall be in its active state for at least 32 SLCK periods |
| 3 | 512_SLCK | PIOBUx shall be in its active state for at least 512 SLCK periods |
| 4 | 4096_SLCK | PIOBUx shall be in its active state for at least 4,096 SLCK periods |
| 5 | 32768_SLCK | PIOBUx shall be in its active state for at least 32,768 SLCK periods |

**Bit 19 – RXLPWKEN** Debug Unit Wakeup Enable

| Value | Description |
|---|---|
| 0 | The Backup RX UART Comparison event has no effect on the Shutdown Controller. |
| 1 | The Backup RX UART Comparison event forces the de-assertion of the SHDN pin. |

**Bit 18 – ACCWKEN** Analog Comparator Controller Wakeup Enable

| Value | Description |
|---|---|
| 0 | The Analog comparator alarm signal has no effect on the Shutdown Controller. |

| Value | Description |
|---|---|
| 1 | The Analog comparator alarm signal forces the de-assertion of the SHDN pin. |

**Bit 17 – RTCWKEN** Real-time Clock Wakeup Enable

| Value | Description |
|---|---|
| 0 | The RTC Alarm signal has no effect on the Shutdown Controller. |
| 1 | The RTC Alarm signal forces the de-assertion of the SHDN pin. |

### 24.7.3 Shutdown Controller Status Register

**Name:** SHDW_SR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | WKUPIS9 | WKUPIS8 |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WKUPIS7 | WKUPIS6 | WKUPIS5 | WKUPIS4 | WKUPIS3 | WKUPIS2 | WKUPIS1 | WKUPIS0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXLPWK | ACCWK | RTCWK | | | | | WKUPS |
| Access | R | R | R | | | | | R |
| Reset | 0 | 0 | 0 | | | | | 0 |

**Bits 16, 17, 18, 19, 20, 21, 22, 23, 24, 25 – WKUPIS** Wakeup 0 to 9 Input Status
0 (DISABLE): The corresponding wakeup input is disabled, or was inactive at the time the debouncer triggered a wakeup event.

1 (ENABLE): The corresponding wakeup input was active at the time the debouncer triggered a wakeup event.

**Bit 7 – RXLPWK** Debug Unit Wakeup

| Value | Description |
|---|---|
| 0 | No wakeup alarm from the Backup RX UART Comparison unit (RXLP) occurred since the last read of SHDW_SR. |
| 1 | At least one wakeup alarm from the Backup RX UART Comparison unit (RXLP) occurred since the last read of SHDW_SR. |

**Bit 6 – ACCWK** Analog Comparator Controller Wakeup

| Value | Description |
|---|---|
| 0 | No wakeup alarm from the ACC occurred since the last read of SHDW_SR. |
| 1 | At least one wakeup alarm from the ACC occurred since the last read of SHDW_SR. |

**Bit 5 – RTCWK** Real-time Clock Wakeup

| Value | Description |
|---|---|
| 0 | No wakeup alarm from the RTC occurred since the last read of SHDW_SR. |
| 1 | At least one wakeup alarm from the RTC occurred since the last read of SHDW_SR. |

**Bit 0 – WKUPS** PIOBU, WKUP Wakeup Status
WKUPIS1 reports the status of the Security Module event.
0 (NO): No wakeup due to the assertion of the PIOBU, WKUP pins has occurred since the last read of SHDW_SR.

1 (PRESENT): At least one wakeup due to the assertion of the PIOBU, WKUP pins has occurred since the last read of SHDW_SR.

| Value | Description |
|---|---|
| 0 | No wakeup alarm from the RTT occurred since the last read of SHDW_SR. |
| 1 | At least one wakeup alarm from the RTT occurred since the last read of SHDW_SR. |

### 24.7.4 Shutdown Controller Wakeup Inputs Register

**Name:** SHDW_WUIR
**Offset:** 0x0C
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | WKUPT9 | WKUPT8 |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WKUPT7 | WKUPT6 | WKUPT5 | WKUPT4 | WKUPT3 | WKUPT2 | WKUPT1 | WKUPT0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | WKUPEN9 | WKUPEN8 |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WKUPEN7 | WKUPEN6 | WKUPEN5 | WKUPEN4 | WKUPEN3 | WKUPEN2 | WKUPEN1 | WKUPEN0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 16, 17, 18, 19, 20, 21, 22, 23, 24, 25 – WKUPT** Wakeup 0 to 9 Input Type
0 (LOW): A falling edge followed by a low level, for a period defined by WKUPDBC, on the corresponding wakeup input forces the wakeup of the core power supply.

1 (HIGH): A rising edge followed by a high level, for a period defined by WKUPDBC, on the corresponding wakeup input forces the wakeup of the core power supply.

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 – WKUPEN** Wakeup 0 to 9 Input Enable
0 (DISABLE): The corresponding wakeup input has no wakeup effect.

1 (ENABLE): The corresponding wakeup input forces the wakeup of the core power supply.

# 25. Periodic Interval Timer (PIT)

## 25.1 Description

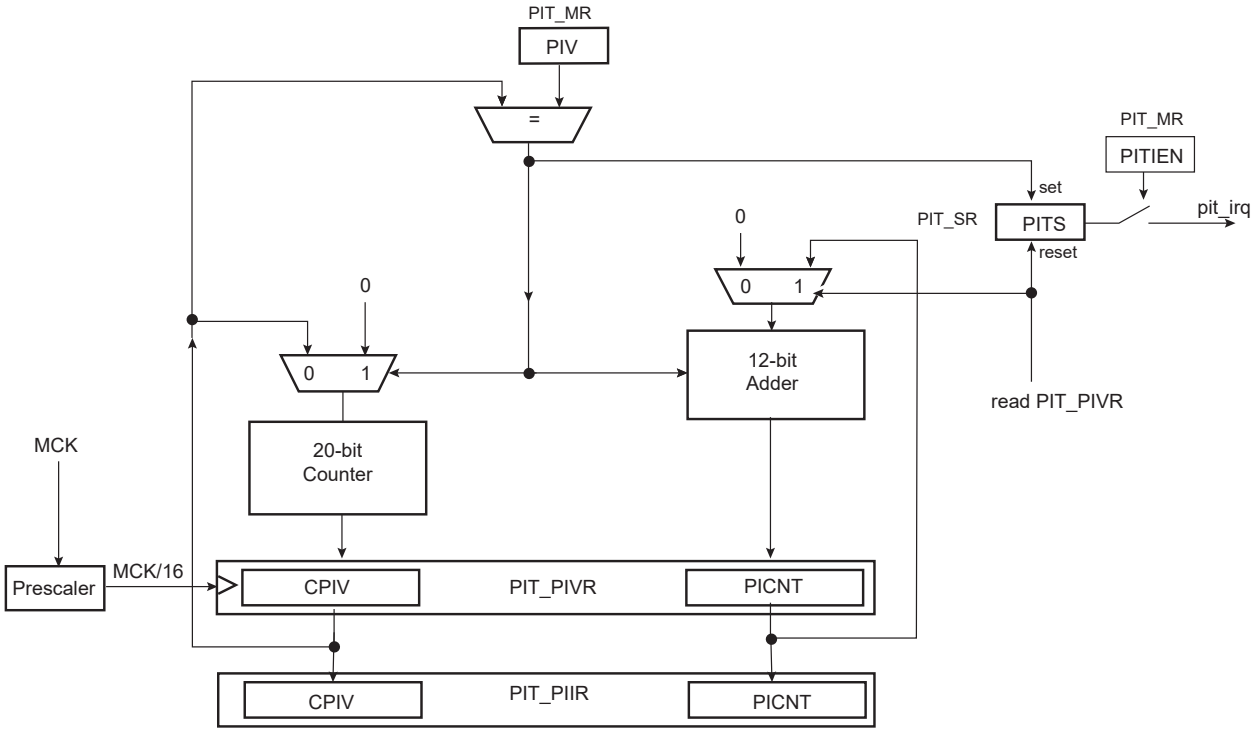
The Periodic Interval Timer (PIT) provides the operating system's scheduler interrupt. It is designed to offer maximum accuracy and efficient management, even for systems with long response time.

## 25.2 Embedded Characteristics

- 20-bit Programmable Counter plus 12-bit Interval Counter
- Reset-on-read Feature
- Both Counters Work on Master Clock/16

## 25.3 Block Diagram

**Figure 25-1. Periodic Interval Timer**




## 25.4 Functional Description

The Periodic Interval Timer provides periodic interrupts for use by operating systems.

The PIT provides a programmable overflow counter and a reset-on-read feature. It is built around two counters: a 20-bit CPIV counter and a 12-bit PICNT counter. Both counters work at Master Clock /16.

The first 20-bit CPIV counter increments from 0 up to a programmable overflow value set in the field PIV of the Mode Register (PIT_MR). When the counter CPIV reaches this value, it resets to 0 and increments

the Periodic Interval Counter, PICNT. The status bit PITS in the Status Register (PIT_SR) rises and triggers an interrupt, provided the interrupt is enabled (PITIEN in PIT_MR).

Writing a new PIV value in PIT_MR does not reset/restart the counters.

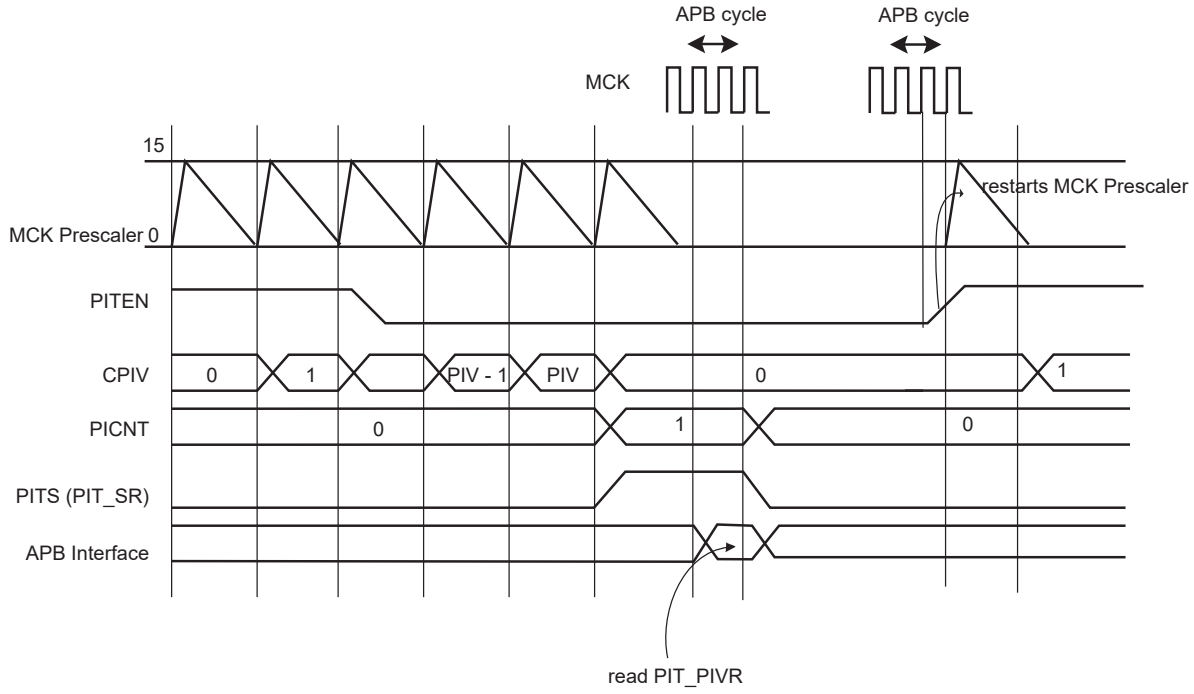
When CPIV and PICNT values are obtained by reading the Periodic Interval Value Register (PIT_PIVR), the overflow counter (PICNT) is reset and the PITS bit is cleared, thus acknowledging the interrupt. The value of PICNT gives the number of periodic intervals elapsed since the last read of PIT_PIVR.

When CPIV and PICNT values are obtained by reading the Periodic Interval Image Register (PIT_PIIR), there is no effect on the counters CPIV and PICNT, nor on the bit PITS. For example, a profiler can read PIT_PIIR without clearing any pending interrupt, whereas a timer interrupt clears the interrupt by reading PIT_PIVR.

The PIT may be enabled/disabled using the PITEN bit in the PIT_MR register (disabled on reset). The PITEN bit only becomes effective when the CPIV value is 0. The figure below illustrates the PIT counting. After the PIT Enable bit is reset (PITEN = 0), the CPIV goes on counting until the PIV value is reached, and is then reset. PIT restarts counting, only if the PITEN is set again.

The PIT is stopped when the core enters debug state.

**Figure 25-2. Enabling/Disabling PIT with PITEN**



## 25.5 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | PIT_MR | 7:0 | PIV[7:0] | | | | | | | |
| | | 15:8 | PIV[15:8] | | | | | | | |
| | | 23:16 | | | | | PIV[19:16] | | | |
| | | 31:24 | | | | | | | PITIEN | PITEN |
| 0x04 | PIT_SR | 7:0 | | | | | | | | PITS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | PIT_PIVR | 7:0 | CPIV[7:0] | | | | | | | |
| | | 15:8 | CPIV[15:8] | | | | | | | |
| | | 23:16 | PICNT[3:0] | | | | CPIV[19:16] | | | |
| | | 31:24 | PICNT[11:4] | | | | | | | |
| 0x0C | PIT_PIIR | 7:0 | CPIV[7:0] | | | | | | | |
| | | 15:8 | CPIV[15:8] | | | | | | | |
| | | 23:16 | PICNT[3:0] | | | | CPIV[19:16] | | | |
| | | 31:24 | PICNT[11:4] | | | | | | | |

### 25.5.1 Periodic Interval Timer Mode Register

**Name:** PIT_MR
**Offset:** 0x00
**Reset:** 0x000FFFFF
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | PITIEN | PITEN |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PIV[19:16] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 1 | 1 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PIV[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PIV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bit 25 – PITIEN** Period Interval Timer Interrupt Enable

| Value | Description |
|---|---|
| 0 | The bit PITS in PIT_SR has no effect on the interrupt. |
| 1 | The bit PITS in PIT_SR asserts an interrupt. |

**Bit 24 – PITEN** Period Interval Timer Enabled

| Value | Description |
|---|---|
| 0 | The Periodic Interval Timer is disabled when the PIV value is reached. |
| 1 | The Periodic Interval Timer is enabled. |

**Bits 19:0 – PIV[19:0]** Periodic Interval Value
Defines the value compared with the primary 20-bit counter of the Periodic Interval Timer (CPIV). The period is equal to (PIV + 1).

### 25.5.2 Periodic Interval Timer Status Register

**Name:** PIT_SR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | PITS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 0 – PITS** Periodic Interval Timer Status

| Value | Description |
|---|---|
| 0 | The Periodic Interval timer has not reached PIV since the last read of PIT_PIVR. |
| 1 | The Periodic Interval timer has reached PIV since the last read of PIT_PIVR. |

### 25.5.3 Periodic Interval Timer Value Register

**Name:** PIT_PIVR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read-only

Reading this register clears PITS in PIT_SR.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PICNT[11:4] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PICNT[3:0] | | | | CPIV[19:16] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CPIV[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CPIV[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:20 – PICNT[11:0]** Periodic Interval Counter
Returns the number of occurrences of periodic intervals since the last read of PIT_PIVR.

**Bits 19:0 – CPIV[19:0]** Current Periodic Interval Value
Returns the current value of the periodic interval timer.

### 25.5.4 Periodic Interval Timer Image Register

**Name:** PIT_PIIR
**Offset:** 0x0C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PICNT[11:4] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PICNT[3:0] | | | | CPIV[19:16] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CPIV[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CPIV[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:20 – PICNT[11:0]** Periodic Interval Counter
Returns the number of occurrences of periodic intervals since the last read of PIT_PIVR.

**Bits 19:0 – CPIV[19:0]** Current Periodic Interval Value
Returns the current value of the periodic interval timer.

# 26. Real-time Clock (RTC)

## 26.1 Description

The Real-time Clock (RTC) peripheral is designed for very low power consumption. For optimal functionality, the RTC requires an accurate external 32.768 kHz clock, which can be provided by a crystal oscillator.

It combines a complete time-of-day clock with alarm and a Gregorian or Persian calendar, complemented by a programmable periodic interrupt. The alarm and calendar registers are accessed by a 32-bit data bus.

The RTC can also be configured for the UTC time format.

The time and calendar values are coded in binary-coded decimal (BCD) format. The time format can be 24-hour mode or 12-hour mode with an AM/PM indicator.

Updating time and calendar fields and configuring the alarm fields are performed by a parallel capture on the 32-bit data bus. An entry control is performed to avoid loading registers with incompatible BCD format data or with an incompatible date according to the current month/year/century.

A clock divider calibration circuitry can be used to compensate for crystal oscillator frequency variations.

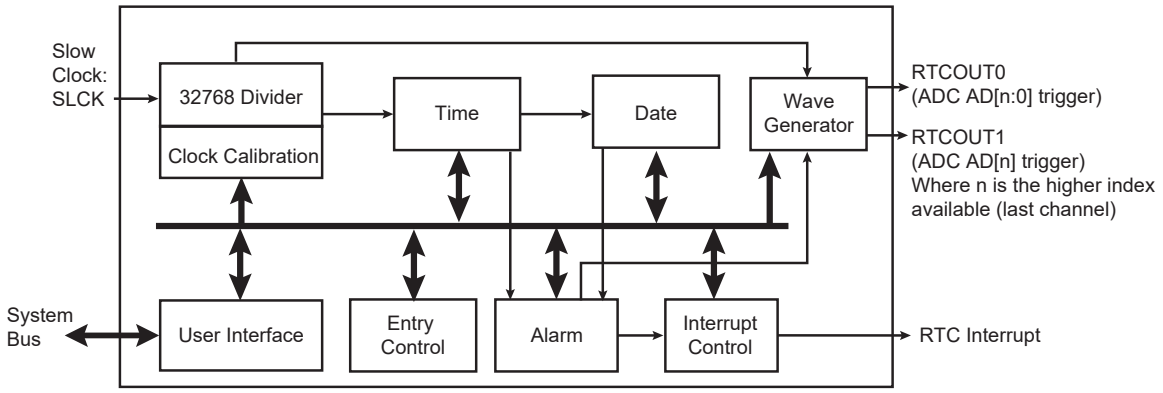
Timestamping capability reports the first and last occurrences of tamper events.

## 26.2 Embedded Characteristics

- Full Asynchronous Design for Ultra Low Power Consumption
- Gregorian, UTC and Persian Modes Supported
- Programmable Periodic Interrupt
- Safety/security Features:
  - Valid Time and Date Programming Check
  - On-The-Fly Time and Date Validity Check
- Counters Calibration Circuitry to Compensate for Crystal Oscillator Variations
- Waveform Generation for Trigger Event
- Tamper Timestamping Registers
- Register Write Protection

## 26.3 Block Diagram

**Figure 26-1. Real-time Clock Block Diagram**




## 26.4 Product Dependencies

### 26.4.1 Power Management

The Real-time Clock is continuously clocked at 32.768 kHz. The Power Management Controller has no effect on RTC behavior.

### 26.4.2 Interrupt

Within the System Controller, the RTC interrupt is OR-wired with all the other module interrupts.

Only one System Controller interrupt line is connected on one of the internal sources of the interrupt controller.

RTC interrupt requires the interrupt controller to be programmed first.

When a System Controller interrupt occurs, the service routine must first determine the cause of the interrupt. This is done by reading each status register of the System Controller peripherals successively.

## 26.5 Functional Description

The RTC provides a full binary-coded decimal (BCD) clock that includes century (19/20), year (with leap years), month, date, day, hours, minutes and seconds reported in RTC Time Register (RTC_TIMR) and RTC Calendar Register (RTC_CALR).

The RTC can operate in UTC mode, giving the number of seconds elapsed since a reference time defined by the user (the UTC standard—ISO 8601—reference time is the 30th of June 1972). In this mode, the timefield is 32 bits wide and coded in hexadecimal format.

The valid year range is up to 2099 in Gregorian mode (or 1300 to 1499 in Persian mode).

The RTC can operate in 24-hour mode or in 12-hour mode with an AM/PM indicator.

Corrections for leap years are included (all years divisible by 4 being leap years except 1900). This is correct up to the year 2099.

The RTC can generate events to trigger ADC measurements.

### 26.5.1 Reference Clock

The reference clock is the Slow Clock (SLCK) which can be driven internally or by an external 32.768 kHz crystal.

During low-power modes of the processor, the oscillator runs and power consumption is critical. The crystal selection must consider the current consumption for power saving and the frequency drift due to temperature effect on the circuit for time accuracy.

### 26.5.2 Timing

In Gregorian and Persian modes, the RTC is updated in real time at one-second intervals in Normal mode for the counters of seconds, at one-minute intervals for the counter of minutes and so on.

In UTC mode, the RTC is updated in real time at one-second intervals (32-bit UTC counter default configuration).

Due to the asynchronous operation of the RTC with respect to the rest of the chip, to be certain that the value read in the RTC registers (century, year, month, date, day, hours, minutes, seconds) are valid and stable, it is necessary to read these registers twice. If the data is the same both times, then it is valid. Therefore, a minimum of two and a maximum of three accesses are required.

### 26.5.3 Alarm

In Gregorian and Persian modes, the RTC has five programmable fields: month, date, hours, minutes and seconds.

Each of these fields can be enabled or disabled to match the alarm condition:

- If all the fields are enabled, an alarm flag is generated (the corresponding flag is asserted and an interrupt generated if enabled) at a given month, date, hour/minute/second.
- If only the “seconds” field is enabled, then an alarm is generated every minute.

Depending on the combination of fields enabled, a large number of possibilities are available to the user ranging from minutes to 365/366 days.

Hour, minute and second matching alarms (SECEN, MINEN, HOUREN) can be enabled independently of SEC, MIN, HOUR fields.

**Note:** To change one of the SEC, MIN, HOUR, DATE, MONTH fields, it is recommended to disable the field before changing the value and then re-enable it after the change has been made. This requires up to three accesses to the RTC_TIMALR or RTC_CALALR. The first access clears the enable corresponding to the field to change (SECEN, MINEN, HOUREN, DATEEN, MTHEN). If the field is already cleared, this access is not required. The second access performs the change of the value (SEC, MIN, HOUR, DATE, MONTH). The third access is required to re-enable the field by writing 1 in SECEN, MINEN, HOUREn, DATEEN, MTHEN fields.

In UTC mode, RTC_TIMALR must be configured to set the UTC alarm value and bit 0 in RTC_CALALR must be used to enable or disable the UTC alarm. If the UTC alarm is enabled, the alarm is generated once the UTC time matches the programmed UTC_TIME alarm field.

To change the UTC_TIME alarm field, proceed as follows:

1. Disable the UTC alarm by clearing the UTCEN bit in RTC_CALALR if it is not already cleared.
2. Change the UTC_TIME alarm value in RTC_TIMALR.
3. Re-enable the UTC alarm by setting the UTCEN bit in RTC_CALALR.

### 26.5.4 Error Checking when Programming

Verification on user interface data is performed when accessing the century, year, month, date, day, hours, minutes, seconds and alarms. A check is performed on illegal BCD entries such as illegal date of the month with regard to the year and century configured.

If one of the time fields is not correct, the data is not loaded into the register/counter and a flag is set in the validity register. The user can not reset this flag. It is reset as soon as an acceptable value is programmed. This avoids any further side effects in the hardware. The same procedure is followed for the alarm.

The following checks are performed:

1. Century (check if it is in range 19–20 or 13–14 in Persian mode)
2. Year (BCD entry check)
3. Date (check range 01–31)
4. Month (check if it is in BCD range 01–12, check validity regarding “date”)
5. Day (check range 1–7)
6. Hour (BCD checks: in 24-hour mode, check range 00–23 and check that AM/PM flag is not set if RTC is set in 24-hour mode; in 12-hour mode check range 01–12)
7. Minute (check BCD and range 00–59)
8. Second (check BCD and range 00–59)

**Note:** If the 12-hour mode is selected by means of the RTC Mode Register (RTC_MR), a 12-hour value can be programmed and the returned value on RTC_TIMR will be the corresponding 24-hour value. The entry control checks the value of the AM/PM indicator (bit 22 of RTC_TIMR) to determine the range to be checked.

**Note:** In UTC mode, no check is performed on the entries. The RTC does not report any failure.

### 26.5.5 RTC Internal Free Running Counter Error Checking

To improve the reliability and security of the RTC, a permanent check is performed on the internal free running counters to report non-BCD or invalid date/time values.

An error is reported by TDERR bit in the status register (RTC_SR) if an incorrect value has been detected. The flag can be cleared by setting the TDERRCLR bit in the Status Clear Command Register (RTC_SCCR).

The TDERR error flag will be set again if the source of the error has not been cleared before clearing the TDERR flag. The clearing of the source of such error can be done by reprogramming a correct value on RTC_CALR and/or RTC_TIMR.

The RTC internal free running counters may automatically clear the source of TDERR due to their roll-over (i.e., every 10 seconds for SECONDS[3:0] field in RTC_TIMR). In this case the TDERR is held high until a clear command is asserted by TDERRCLR bit in RTC_SCCR.

### 26.5.6 Updating Time/Calendar

#### 26.5.6.1 Gregorian and Persian Modes

The update of the time/calendar must be synchronized on a second periodic event by either polling the RTC_SR.SEC status bit or by enabling the SECEN interrupt in the RTC_IER register.

Once the second event occurs, the user must stop the RTC by setting the corresponding field in the Control Register (RTC_CR). Bit UPDTIM must be set to update time fields (hour, minute, second) and bit UPDCAL must be set to update calendar fields (century, year, month, date, day).

The ACKUPD bit must then be read to 1 by either polling the RTC_SR or by enabling the ACKUPD interrupt in the RTC_IER. Once ACKUPD is read to 1, it is mandatory to clear this flag by writing the corresponding bit in the RTC_SCCR, after which the user can write to the Time Register, the Calendar Register, or both.

Once the update is finished, the user must write UPDTIM and/or UPDCAL to 0 in the RTC_CR.

The timing sequence of the time/calendar update is described in the figure below.

When entering the programming mode of the calendar fields, the time fields remain enabled and both the time and the calendar fields are stopped. This is due to the location of the calendar logical circuity (downstream for low-power considerations). It is highly recommended to prepare all the fields to be updated before entering programming mode. In successive update operations, the user must wait for at least one second after resetting the UPDTIM/UPDCAL bit in the RTC_CR before setting these bits again. This is done by waiting for the SEC flag in the RTC_SR before setting the UPDTIM/UPDCAL bit. After resetting UPDTIM/UPDCAL, the SEC flag must also be cleared.

**Figure 26-2. Time/Calendar Update Timing Diagram**

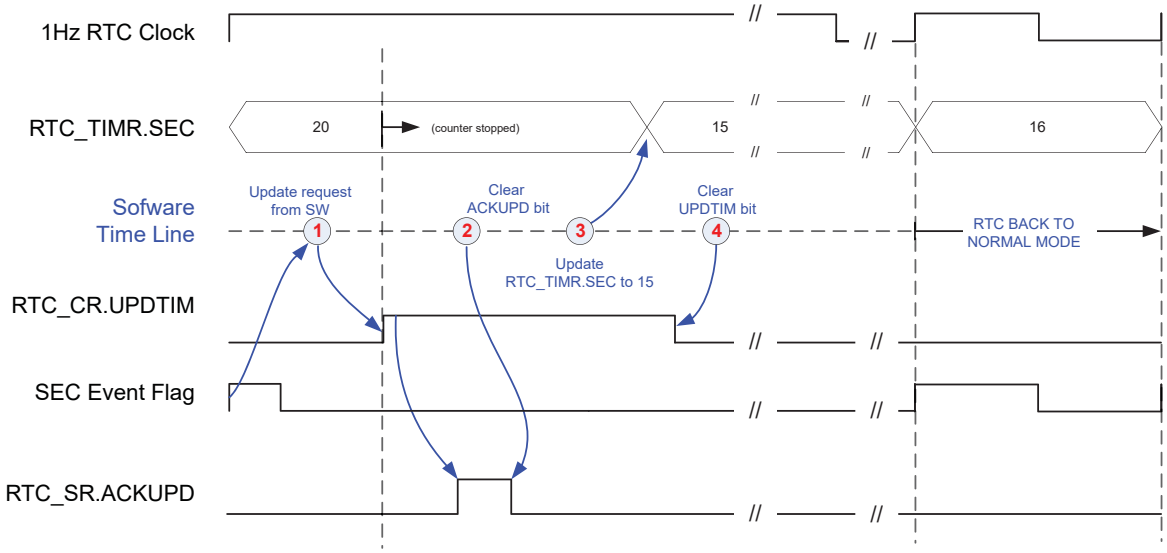
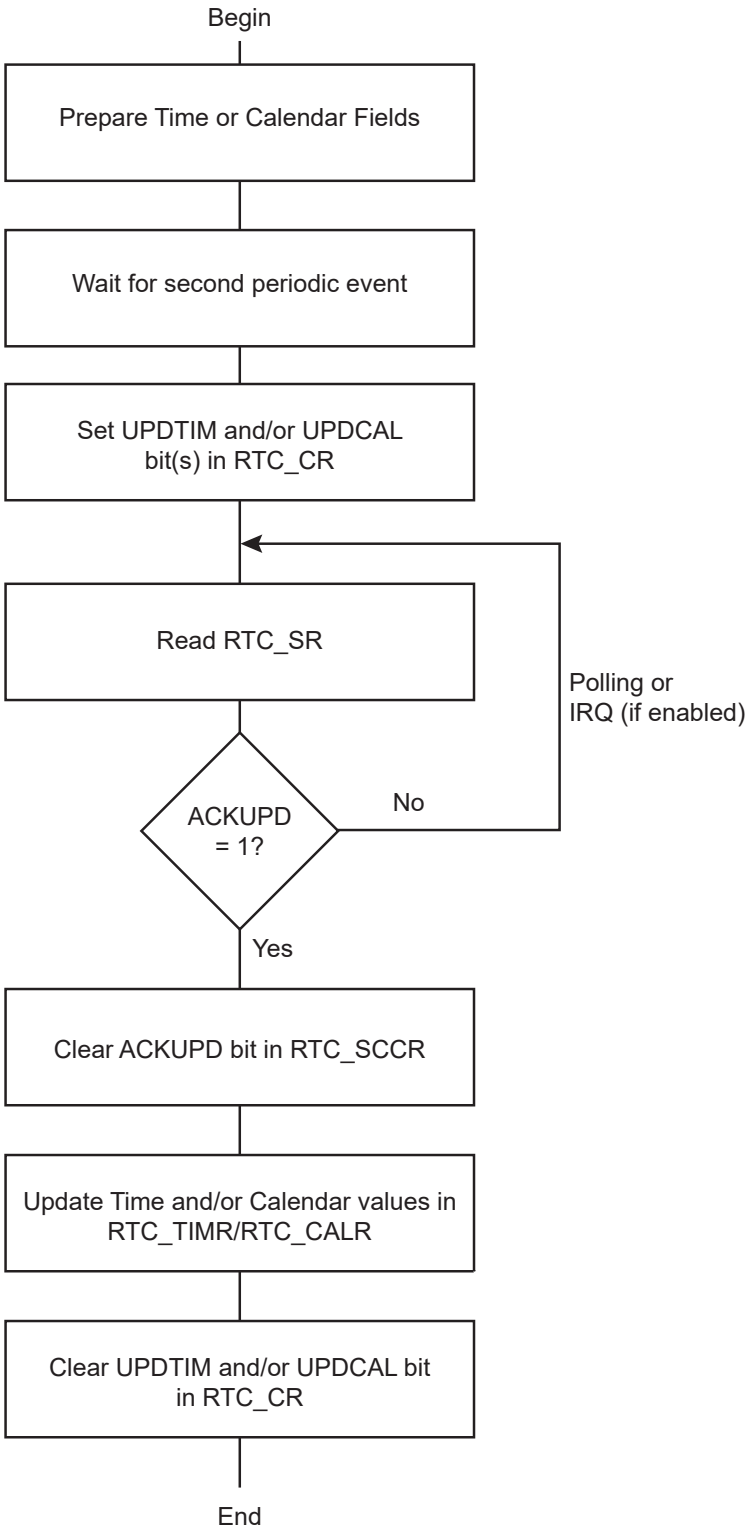


**Figure 26-3. Gregorian and Persian Modes Update Sequence**




#### 26.5.6.2 UTC Mode

The update of the UTC time field must be synchronized on a second periodic event by either polling the RTC_SR.SEC status bit or by enabling the SECEN interrupt in the RTC_IER.

Once the second event occurs, the user must stop the RTC by setting the UPDTIM field in the Control Register (RTC_CR).

The ACKUPD bit must then be read to 1 by either polling the RTC_SR or by enabling the ACKUPD interrupt in the RTC_IER. Once ACKUPD is read to 1, it is mandatory to clear this flag by writing the corresponding bit in the RTC_SCCR, after which the user can write to the Time Register.

Once the update is finished, the user must write UPDTIM to 0 in the RTC_CR.

The timing sequence of the UTC time update is described in the figure below.

In successive update operations, the user must wait for at least one second after resetting the UPDTIM bit in the RTC_CR before setting this bit again. This is done by waiting for the SEC flag in the RTC_SR before setting UPDTIM bit. After resetting UPDTIM, the SEC flag must also be cleared.

**Figure 26-4. UTC Time Update Timing Diagram**

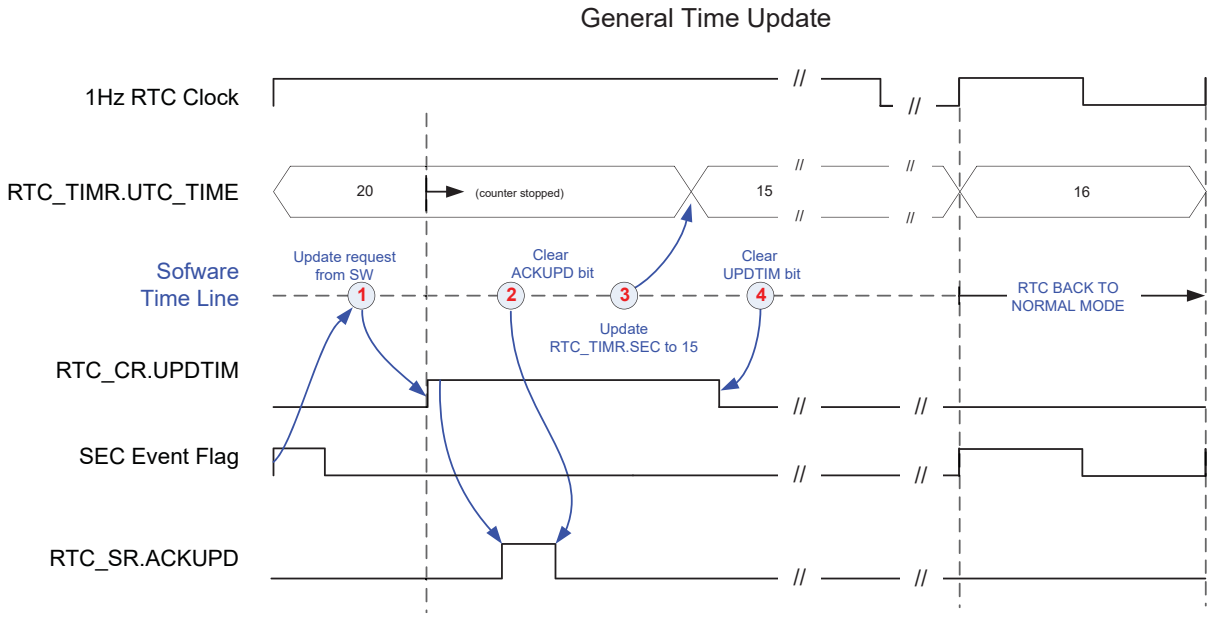
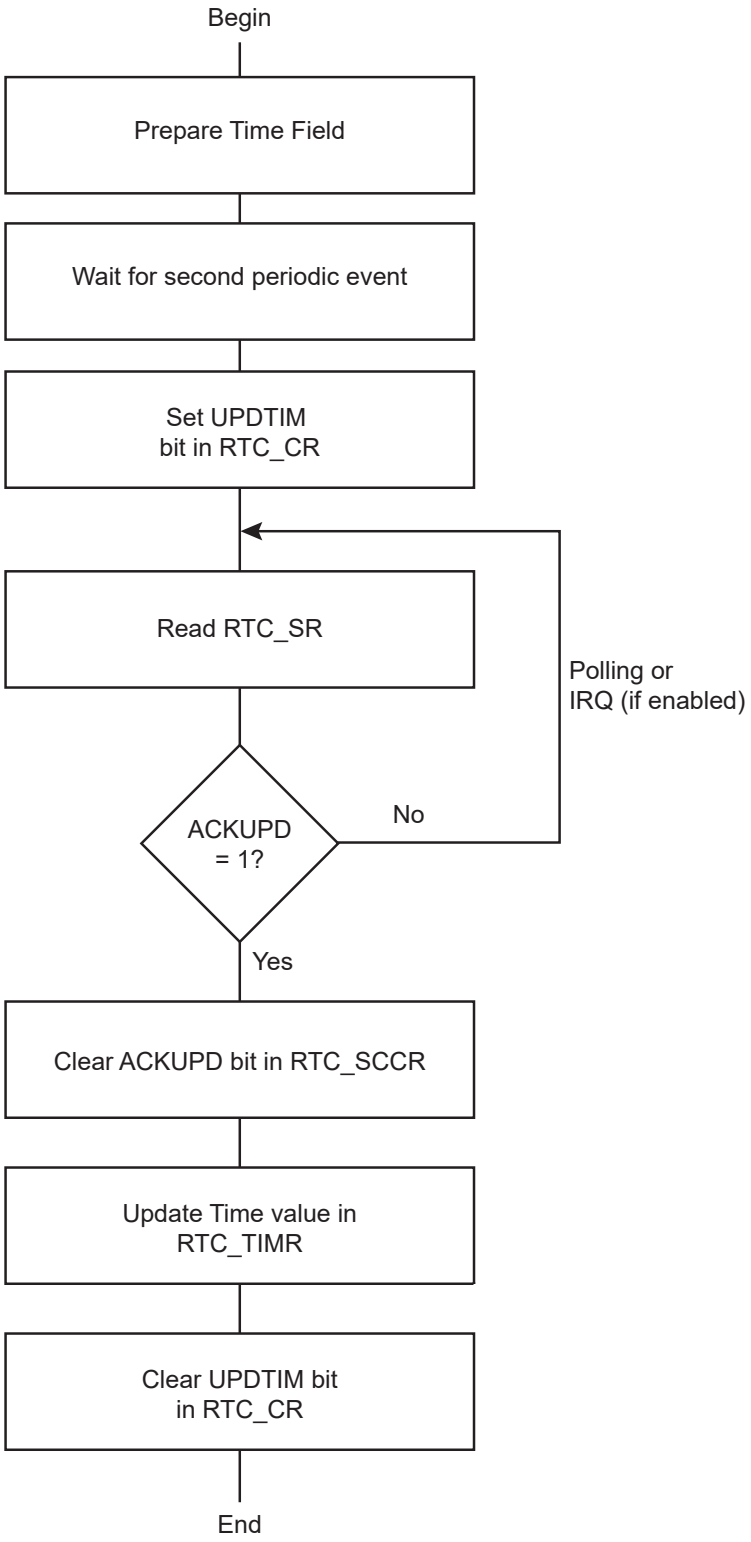


**Figure 26-5. UTC Mode Update Sequence**




### 26.5.7 RTC Accurate Clock Calibration

The crystal oscillator that drives the RTC may not be as accurate as expected mainly due to temperature variation. The RTC is equipped with circuitry able to correct slow clock crystal drift.

To compensate for possible temperature variations over time, this accurate clock calibration circuitry can be programmed on-the-fly and also programmed during application manufacturing, in order to correct the crystal frequency accuracy at room temperature (20–25°C). The typical clock drift range at room temperature is ±20 ppm.

In the device operating temperature range, the 32.768 kHz crystal oscillator clock inaccuracy can be up to -200 ppm.

The RTC clock calibration circuitry allows positive or negative correction in a range of 1.5 ppm to 1950 ppm.

The calibration circuitry is fully digital. Thus, the configured correction is independent of temperature, voltage, process, etc., and no additional measurement is required to check that the correction is effective.

If the correction value configured in the calibration circuitry results from an accurate crystal frequency measure, the remaining accuracy is bounded by the values listed below:

- Below 1 ppm, for an initial crystal drift between 1.5 ppm up to 20 ppm, and from 30 ppm to 90 ppm
- Below 2 ppm, for an initial crystal drift between 20 ppm up to 30 ppm, and from 90 ppm to 130 ppm
- Below 5 ppm, for an initial crystal drift between 130 ppm up to 200 ppm

The calibration circuitry does not modify the 32.768 kHz crystal oscillator clock frequency but it acts by slightly modifying the 1 Hz clock period from time to time. The correction event occurs every 1 + [(20 - (19 x HIGHPPM)) x CORRECTION] seconds. When the period is modified, depending on the sign of the correction, the 1 Hz clock period increases or reduces by around 4 ms. Depending on the CORRECTION, NEGPPM and HIGHPPM values configured in RTC_MR, the period interval between two correction events differs.

**Figure 26-6. Calibration Circuitry**

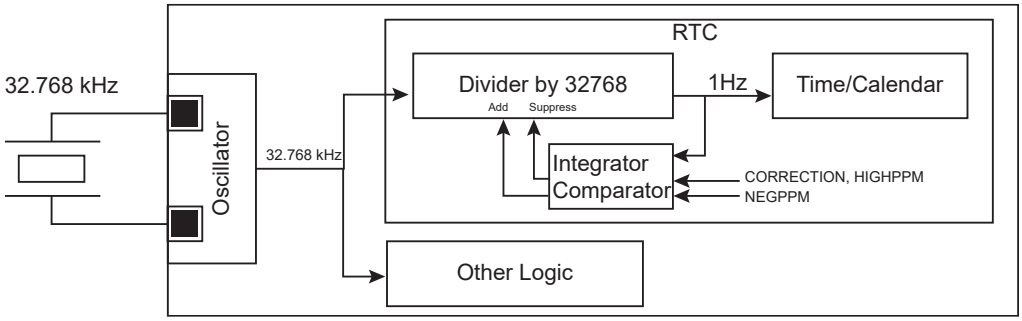
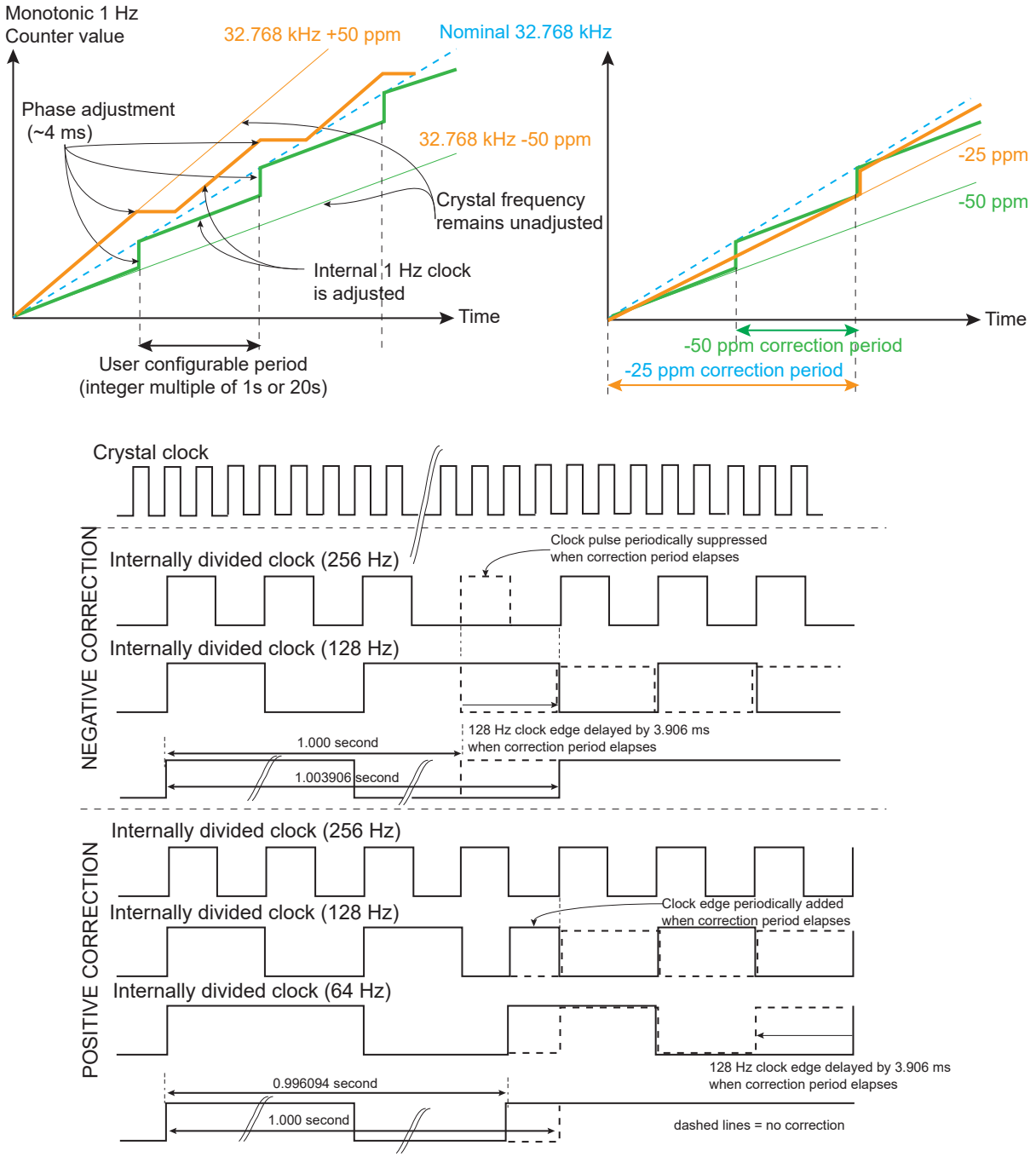


**Figure 26-7. Calibration Circuitry Waveforms**




The inaccuracy of a crystal oscillator at typical room temperature (±20 ppm at 20–25 °C) can be compensated if a reference clock/signal is used to measure such inaccuracy. This kind of calibration operation can be set up during the final product manufacturing by means of measurement equipment embedding such a reference clock. The correction of value must be programmed into the (RTC_MR), and this value is kept as long as the circuitry is powered (backup area). Removing the backup power supply cancels this calibration. This room temperature calibration can be further processed by means of the networking capability of the target application.

Note that this adjustment does not take into account the temperature variation.

The frequency drift (up to -200 ppm) due to temperature variation can be compensated using a reference time if the application can access such a reference. If a reference time cannot be used, a temperature sensor can be placed close to the crystal oscillator in order to get the operating temperature of the crystal oscillator. Once obtained, the temperature may be converted using a lookup table (describing the accuracy/temperature curve of the crystal oscillator used) and RTC_MR configured accordingly. The calibration can be performed on-the-fly. This adjustment method is not based on a measurement of the crystal frequency/drift and therefore can be improved by means of the networking capability of the target application.

If no crystal frequency adjustment has been done during manufacturing, it is still possible to do it. In the case where a reference time of the day can be obtained through LAN/WAN network, it is possible to calculate the drift of the application crystal oscillator by comparing the values read on RTC Time Register (RTC_TIMR) and programming the HIGHPPM and CORRECTION fields on RTC_MR according to the difference measured between the reference time and those of RTC_TIMR.

### 26.5.8 Waveform Generation

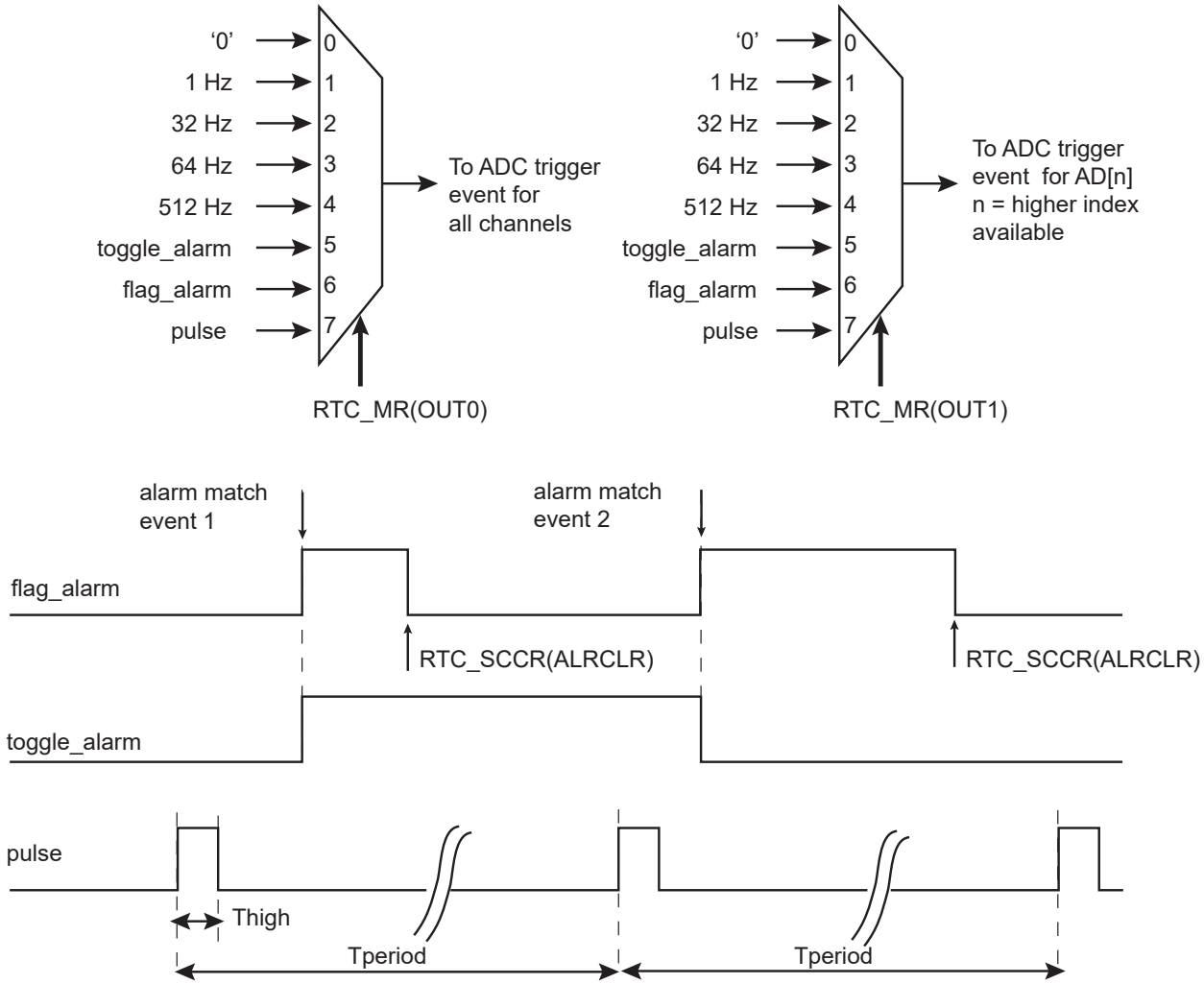
Waveforms can be generated in order to take advantage of the RTC inherent prescalers while the RTC is the only powered circuitry (Low-power mode of operation, Backup mode) or in any active mode. Entering Backup or Low-power operating modes does not affect the waveform generation outputs.

The RTC waveforms are internally routed to ADC trigger events. These events can be configured to provide several types of waveforms. The figure below illustrates the different signals available to generate the waveforms. Two different triggers can be generated at a time. The first is configured in RTC_MR.OUT0 while the second is configurable in RTC_MR.OUT1. OUT0 manages the trigger for channel AD[n:0] (where n is the higher index available (last channel)), while OUT1 manages the channel AD[n] only for specific modes. See the section "Analog to Digital Converter (ADC)" for selection of the measurement triggers and associated modes of operation.

The first selection choice sticks the associated output at 0. (This is the reset value and it can be used at any time to disable the waveform generation).

Selection choices 1 to 4 respectively select 1 Hz, 32 Hz, 64 Hz and 512 Hz.

**Figure 26-8. Waveform Generation for ADC Trigger Event**




### 26.5.9 Tamper Timestamping

As soon as a tamper is detected, the tamper counter is incremented and the RTC stores the time of the day, the date and the source of the tamper event in registers located in the backup area. Up to two tamper events can be stored.

In UTC mode, only the UTC time is stored. The date information is not relevant.

The tamper counter saturates at 15. Once this limit is reached, the exact number of tamper occurrences since the last read of stamping registers cannot be known.

The first set of timestamping registers (RTC_TSTR0, RTC_TSDR0, RTC_TSSR0) cannot be overwritten, so once they have been written all data are stored until the registers are reset. Therefore these registers are storing the first tamper occurrence after a read.

The second set of timestamping registers (RTC_TSTR1, RTC_TSDR1, RTC_TSSR1) are overwritten each time a tamper event is detected. Thus the date and the time data of the first and the second stamping registers may be equal. This occurs when the tamper counter value carried on field TEVCNT in RTC_TSTR0 equals 1. Thus this second set of registers stores the last occurrence of tamper before a read.

Reading a set of timestamping registers requires three accesses, one for the time of the day, one for the date and one for the tamper source.

Reading the third part (RTC_TSSR0/1) of a timestamping register set clears the whole content of the registers (time, date and tamper source) and makes the timestamping registers available to store a new event.

## 26.6 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | RTC_CR | 7:0 | | | | | | | UPDCAL | UPDTIM |
| | | 15:8 | | | | | | | TIMEVSEL[1:0] | |
| | | 23:16 | | | | | | | CALEVSEL[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x04 | RTC_MR | 7:0 | | | | NEGPPM | | UTC | PERSIAN | HRMOD |
| | | 15:8 | HIGHPPM | CORRECTION[6:0] | | | | | | |
| | | 23:16 | | OUT1[2:0] | | | | OUT0[2:0] | | |
| | | 31:24 | | | TPERIOD[1:0] | | | THIGH[2:0] | | |
| 0x08 | RTC_TIMR | 7:0 | | SEC[6:0] | | | | | | |
| | | 15:8 | | MIN[6:0] | | | | | | |
| | | 23:16 | | AMPM | HOUR[5:0] | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | RTC_TIMR (UTC_MODE) | 7:0 | UTC_TIME[7:0] | | | | | | | |
| | | 15:8 | UTC_TIME[15:8] | | | | | | | |
| | | 23:16 | UTC_TIME[23:16] | | | | | | | |
| | | 31:24 | UTC_TIME[31:24] | | | | | | | |
| 0x0C | RTC_CALR | 7:0 | | CENT[6:0] | | | | | | |
| | | 15:8 | YEAR[7:0] | | | | | | | |
| | | 23:16 | DAY[2:0] | | | MONTH[4:0] | | | | |
| | | 31:24 | | | DATE[5:0] | | | | | |
| 0x10 | RTC_TIMALR | 7:0 | SECEN | SEC[6:0] | | | | | | |
| | | 15:8 | MINEN | MIN[6:0] | | | | | | |
| | | 23:16 | HOUREN | AMPM | HOUR[5:0] | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | RTC_TIMALR (UTC_MODE) | 7:0 | UTC_TIME[7:0] | | | | | | | |
| | | 15:8 | UTC_TIME[15:8] | | | | | | | |
| | | 23:16 | UTC_TIME[23:16] | | | | | | | |
| | | 31:24 | UTC_TIME[31:24] | | | | | | | |
| 0x14 | RTC_CALALR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | MTHEN | | | MONTH[4:0] | | | | |
| | | 31:24 | DATEEN | | DATE[5:0] | | | | | |
| 0x14 | RTC_CALALR (UTC_MODE) | 7:0 | | | | | | | | UTCEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | RTC_SR | 7:0 | | | TDERR | CALEV | TIMEV | SEC | ALARM | ACKUPD |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x1C | RTC_SCCR | 7:0 | | | TDERRCLR | CALCLR | TIMCLR | SECCLR | ALRCLR | ACKCLR | |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x20 | RTC_IER | 7:0 | | | TDERREN | CALEN | TIMEN | SECEN | ALREN | ACKEN | |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x24 | RTC_IDR | 7:0 | | | TDERRDIS | CALDIS | TIMDIS | SECDIS | ALRDIS | ACKDIS | |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x28 | RTC_IMR | 7:0 | | | TDERR | CAL | TIM | SEC | ALR | ACK | |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x2C | RTC_VER | 7:0 | | | | | NVCALALR | NVTIMALR | NVCAL | NVTIM | |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x30 ... 0xAF | Reserved | | | | | | | | | | |
| 0xB0 | RTC_TSTR0 | 7:0 | | SEC[6:0] | | | | | | | |
| | | 15:8 | | MIN[6:0] | | | | | | | |
| | | 23:16 | | AMPM | HOUR[5:0] | | | | | | |
| | | 31:24 | BACKUP | | | | TEVCNT[3:0] | | | | |
| 0xB0 | RTC_TSTR0 (UTC_MODE) | 7:0 | | | | | | | | | |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | BACKUP | | | | TEVCNT[3:0] | | | | |
| 0xB4 | RTC_TSDR0 | 7:0 | | CENT[6:0] | | | | | | | |
| | | 15:8 | YEAR[7:0] | | | | | | | | |
| | | 23:16 | DAY[2:0] | | | MONTH[4:0] | | | | | |
| | | 31:24 | | | DATE[5:0] | | | | | | |
| 0xB4 | RTC_TSDRx (UTC_MODE) | 7:0 | UTC_TIME[7:0] | | | | | | | | |
| | | 15:8 | UTC_TIME[15:8] | | | | | | | | |
| | | 23:16 | UTC_TIME[23:16] | | | | | | | | |
| | | 31:24 | UTC_TIME[31:24] | | | | | | | | |
| 0xB8 | RTC_TSSR0 | 7:0 | | | | | JTAG | TST | | | |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 | |
| | | 31:24 | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xBC | RTC_TSTR1 | 7:0 | | SEC[6:0] | | | | | | |
| | | 15:8 | | MIN[6:0] | | | | | | |
| | | 23:16 | | AMPM | HOUR[5:0] | | | | | |
| | | 31:24 | BACKUP | | | | | | | |
| 0xBC | RTC_TSTR1 (UTC_MODE) | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | BACKUP | | | | | | | |
| 0xC0 | RTC_TSDR1 | 7:0 | | CENT[6:0] | | | | | | |
| | | 15:8 | YEAR[7:0] | | | | | | | |
| | | 23:16 | DAY[2:0] | | | MONTH[4:0] | | | | |
| | | 31:24 | | | DATE[5:0] | | | | | |
| 0xC4 | RTC_TSSR1 | 7:0 | | | | | JTAG | TST | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| | | 31:24 | | | | | | | | |

### 26.6.1 RTC Control Register

**Name:** RTC_CR
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CALEVSEL[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | TIMEVSEL[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | UPDCAL | UPDTIM |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bits 17:16 – CALEVSEL[1:0]** Calendar Event Selection
The event that generates the flag CALEV in RTC_SR depends on the value of CALEVSEL

In UTC mode, this field has no effect on the RTC_SR.

| Value | Name | Description |
|---|---|---|
| 0 | WEEK | Week change (every Monday at time 00:00:00) |
| 1 | MONTH | Month change (every 01 of each month at time 00:00:00) |
| 2 | YEAR | Year change (every January 1 at time 00:00:00) |
| 3 | YEAR | Reserved |

**Bits 9:8 – TIMEVSEL[1:0]** Time Event Selection
The event that generates the flag TIMEV in RTC_SR depends on the value of TIMEVSEL.

In UTC mode, this field has no effect on the RTC_SR.

| Value | Name | Description |
|---|---|---|
| 0 | MINUTE | Minute change |
| 1 | HOUR | Hour change |
| 2 | MIDNIGHT | Every day at midnight |
| 3 | NOON | Every day at noon |

**Bit 1 – UPDCAL** Update Request Calendar Register
Calendar counting consists of day, date, month, year and century counters. Calendar counters can be programmed once this bit is set and acknowledged by the bit ACKUPD of the RTC_SR.

In UTC mode, this bit has no effect on the RTC behavior.

| Value | Description |
|---|---|
| 0 | No effect or, if UPDCAL has been previously written to 1, stops the update procedure. |
| 1 | Stops the RTC calendar counting. |

**Bit 0 – UPDTIM** Update Request Time Register
Time counting consists of second, minute and hour counters. Time counters can be programmed once this bit is set and acknowledged by the bit ACKUPD of the RTC_SR.

| Value | Description |
|---|---|
| 0 | No effect or, if UPDTIM has been previously written to 1, stops the update procedure. |
| 1 | Stops the RTC time counting. |

### 26.6.2 RTC Mode Register

**Name:** RTC_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | TPERIOD[1:0] | | | THIGH[2:0] | | |
| Access | | | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | 0 | 0 | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | OUT1[2:0] | | | | OUT0[2:0] | | |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | HIGHPPM | CORRECTION[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | NEGPPM | | UTC | PERSIAN | HRMOD |
| Access | | | | R/W | | R/W | R/W | R/W |
| Reset | | | | 0 | | 0 | 0 | 0 |

**Bits 29:28 – TPERIOD[1:0]** Period of the Output Pulse

| Value | Name | Description |
|---|---|---|
| 0 | P_1S | 1 second |
| 1 | P_500MS | 500 ms |
| 2 | P_250MS | 250 ms |
| 3 | P_125MS | 125 ms |

**Bits 26:24 – THIGH[2:0]** High Duration of the Output Pulse

| Value | Name | Description |
|---|---|---|
| 0 | H_31MS | 31.2 ms |
| 1 | H_16MS | 15.6 ms |
| 2 | H_4MS | 3.91 ms |
| 3 | H_976US | 976 µs |
| 4 | H_488US | 488 µs |
| 5 | H_122US | 122 µs |
| 6 | H_30US | 30.5 µs |
| 7 | H_15US | 15.2 µs |

**Bits 22:20 – OUT1[2:0]** ADC Last Channel Trigger Event Source Selection

| Value | Name | Description |
|---|---|---|
| 0 | NO_WAVE | No waveform, stuck at ‘0’ |
| 1 | FREQ1HZ | 1 Hz square wave |
| 2 | FREQ32HZ | 32 Hz square wave |
| 3 | FREQ64HZ | 64 Hz square wave |
| 4 | FREQ512HZ | 512 Hz square wave |
| 5 | ALARM_TOGGLE | Output toggles when alarm flag rises |
| 6 | ALARM_FLAG | Output is a copy of the alarm flag |
| 7 | PROG_PULSE | Duty cycle programmable pulse |

**Bits 18:16 – OUT0[2:0]** All ADC Channel Trigger Event Source Selection

| Value | Name | Description |
|---|---|---|
| 0 | NO_WAVE | No waveform, stuck at ‘0’ |
| 1 | FREQ1HZ | 1 Hz square wave |
| 2 | FREQ32HZ | 32 Hz square wave |
| 3 | FREQ64HZ | 64 Hz square wave |
| 4 | FREQ512HZ | 512 Hz square wave |
| 5 | ALARM_TOGGLE | Output toggles when alarm flag rises |
| 6 | ALARM_FLAG | Output is a copy of the alarm flag |
| 7 | PROG_PULSE | Duty cycle programmable pulse |

**Bit 15 – HIGHPPM** HIGH PPM Correction
If the absolute value of the correction to be applied is lower than 30 ppm, it is recommended to clear HIGHPPM. HIGHPPM set to 1 is recommended for 30 ppm correction and above.

**Formula:**

If HIGHPPM = 0, then the clock frequency correction range is from 1.5 ppm up to 98 ppm. The RTC accuracy is less than 1 ppm for a range correction from 1.5 ppm up to 30 ppm.

The correction field must be programmed according to the required correction in ppm; the formula is as follows:

$$\mathrm{CORRECTION} = \frac{3906}{20 \times \mathrm{ppm}} - 1$$

The value obtained must be rounded to the nearest integer prior to being programmed into CORRECTION field.

If HIGHPPM = 1, then the clock frequency correction range is from 30.5 ppm up to 1950 ppm. The RTC accuracy is less than 1 ppm for a range correction from 30.5 ppm up to 90 ppm.

The correction field must be programmed according to the required correction in ppm; the formula is as follows:

$$\mathrm{CORRECTION} = \frac{3906}{\mathrm{ppm}} - 1$$

The value obtained must be rounded to the nearest integer prior to be programmed into CORRECTION field.

If NEGPPM is set to 1, the ppm correction is negative (used to correct crystals that are faster than the nominal 32.768 kHz).

| Value | Description |
|---|---|
| 0 | Lower range ppm correction with accurate correction. |
| 1 | Higher range ppm correction with accurate correction. |

**Bits 14:8 – CORRECTION[6:0]** Slow Clock Correction

| Value | Description |
|---|---|
| 0 | No correction |
| 1-127 | The slow clock will be corrected according to the formula given in HIGHPPM description. |

**Bit 4 – NEGPPM** Negative PPM Correction
See CORRECTION and HIGHPPM field descriptions.

NEGPPM must be cleared to correct a crystal slower than 32.768 kHz.

| Value | Description |
|---|---|
| 0 | Positive correction (the divider will be slightly higher than 32768). |
| 1 | Negative correction (the divider will be slightly lower than 32768). |

**Bit 2 – UTC** UTC Time Format
It is forbidden to write a one to the UTC and PERSIAN bits at the same time.

| Value | Description |
|---|---|
| 0 | Gregorian or Persian calendar. |
| 1 | UTC format. |

**Bit 1 – PERSIAN** PERSIAN Calendar

| Value | Description |
|---|---|
| 0 | Gregorian calendar. |
| 1 | Persian calendar. |

**Bit 0 – HRMOD** 12-/24-hour Mode

| Value | Description |
|---|---|
| 0 | 24-hour mode is selected. |
| 1 | 12-hour mode is selected. |

### 26.6.3 RTC Time Register

**Name:** RTC_TIMR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read/Write

In UTC mode, this register view is not relevant, see 26.6.7 RTC_TIMALR (UTC_MODE) .

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | AMPM | HOUR[5:0] | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | MIN[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | SEC[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 22 – AMPM** Ante Meridiem Post Meridiem Indicator
This bit is the AM/PM indicator in 12-hour mode.

| Value | Description |
|---|---|
| 0 | AM. |
| 1 | PM. |

**Bits 21:16 – HOUR[5:0]** Current Hour
The range that can be set is 1–12 (BCD) in 12-hour mode or 0–23 (BCD) in 24-hour mode.

**Bits 14:8 – MIN[6:0]** Current Minute
The range that can be set is 0–59 (BCD).

The lowest four bits encode the units. The higher bits encode the tens.

**Bits 6:0 – SEC[6:0]** Current Second
The range that can be set is 0–59 (BCD).

The lowest four bits encode the units. The higher bits encode the tens.

### 26.6.4 RTC Time Register (UTC_MODE)

**Name:** RTC_TIMR (UTC_MODE)
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read/Write

This configuration is relevant only if UTC = 1 in RTC_MR.

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | UTC_TIME[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UTC_TIME[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UTC_TIME[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UTC_TIME[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – UTC_TIME[31:0]** Current UTC Time
Any value can be set.

### 26.6.5 RTC Calendar Register

**Name:** RTC_CALR
**Offset:** 0x0C
**Reset:** 0x01E11220
**Property:** Read/Write

In UTC mode, values read in this register are not relevant.

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | DATE[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DAY[2:0] | | | MONTH[4:0] | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | YEAR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | CENT[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:24 – DATE[5:0]** Current Day in Current Month
The range that can be set is 01–31 (BCD).

The lowest four bits encode the units. The higher bits encode the tens.

**Bits 23:21 – DAY[2:0]** Current Day in Current Week
The range that can be set is 1–7 (BCD).

The coding of the number (which number represents which day) is user-defined as it has no effect on the date counter.

**Bits 20:16 – MONTH[4:0]** Current Month
The range that can be set is 01–12 (BCD).

The lowest four bits encode the units. The higher bits encode the tens.

**Bits 15:8 – YEAR[7:0]** Current Year
The range that can be set is 00–99 (BCD).

The lowest four bits encode the units. The higher bits encode the tens.

**Bits 6:0 – CENT[6:0]** Current Century
The range that can be set is 19–20 (Gregorian) or 13–14 (Persian) (BCD).

The lowest four bits encode the units. The higher bits encode the tens.

### 26.6.6 RTC Time Alarm Register

**Name:** RTC_TIMALR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

To change one of the SEC, MIN, HOUR fields, it is recommended to disable the field before changing the value and then re-enable it after the change has been made. This requires up to three accesses to the RTC_TIMALR. The first access clears the enable corresponding to the field to change (SECEN, MINEN, HOUREN). If the field is already cleared, this access is not required. The second access performs the change of the value (SEC, MIN, HOUR). The third access is required to re-enable the field by writing 1 in SECEN, MINEN, HOUREN fields.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | HOUREN | AMPM | HOUR[5:0] | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MINEN | MIN[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SECEN | SEC[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 23 – HOUREN** Hour Alarm Enable

| Value | Description |
|---|---|
| 0 | The hour-matching alarm is disabled. |
| 1 | The hour-matching alarm is enabled. |

**Bit 22 – AMPM** AM/PM Indicator
This field is the alarm field corresponding to the BCD-coded hour counter.

**Bits 21:16 – HOUR[5:0]** Hour Alarm
This field is the alarm field corresponding to the BCD-coded hour counter.

**Bit 15 – MINEN** Minute Alarm Enable

| Value | Description |
|---|---|
| 0 | The minute-matching alarm is disabled. |
| 1 | The minute-matching alarm is enabled. |

**Bits 14:8 – MIN[6:0]** Minute Alarm
This field is the alarm field corresponding to the BCD-coded minute counter.

**Bit 7 – SECEN** Second Alarm Enable

| Value | Description |
|---|---|
| 0 | The second-matching alarm is disabled. |
| 1 | The second-matching alarm is enabled. |

**Bits 6:0 – SEC[6:0]** Second Alarm
This field is the alarm field corresponding to the BCD-coded second counter.

### 26.6.7 RTC Time Alarm Register (UTC_MODE)

**Name:** RTC_TIMALR (UTC_MODE)
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read/Write

This configuration is relevant only if UTC = 1 in RTC_MR.

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | UTC_TIME[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UTC_TIME[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UTC_TIME[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UTC_TIME[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – UTC_TIME[31:0]** UTC_TIME Alarm
This field is the alarm field corresponding to the UTC time counter. To change it, proceed as follows:

1. Disable the UTC alarm by clearing the UTCEN bit in RTC_CALALR if it is not already cleared.
2. Change the UTC_TIME alarm value.
3. Enable the UTC alarm by setting the UTCEN bit in RTC_CALALR.

### 26.6.8 RTC Calendar Alarm Register

**Name:** RTC_CALALR
**Offset:** 0x14
**Reset:** 0x01010000
**Property:** Read/Write

In UTC mode, this register view is not relevant, see 26.6.9 RTC_CALALR (UTC_MODE).

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

To change one of the DATE, MONTH fields, it is recommended to disable the field before changing the value and then re-enable it after the change has been made. This requires up to three accesses to the RTC_CALALR. The first access clears the enable corresponding to the field to change (DATEEN, MTHEN). If the field is already cleared, this access is not required. The second access performs the change of the value (DATE, MONTH). The third access is required to re-enable the field by writing 1 in DATEEN, MTHEN fields.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATEEN | | DATE[5:0] | | | | | |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MTHEN | | | MONTH[4:0] | | | | |
| Access | R/W | | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | | 0 | 0 | 0 | 0 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 31 – DATEEN** Date Alarm Enable

| Value | Description |
|---|---|
| 0 | The date-matching alarm is disabled. |
| 1 | The date-matching alarm is enabled. |

**Bits 29:24 – DATE[5:0]** Date Alarm
This field is the alarm field corresponding to the BCD-coded date counter.

**Bit 23 – MTHEN** Month Alarm Enable

| Value | Description |
|---|---|
| 0 | The month-matching alarm is disabled. |
| 1 | The month-matching alarm is enabled. |

**Bits 20:16 – MONTH[4:0]** Month Alarm
This field is the alarm field corresponding to the BCD-coded month counter.

### 26.6.9 RTC Calendar Alarm Register (UTC_MODE)

**Name:** RTC_CALALR (UTC_MODE)
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | UTCEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – UTCEN** UTC Alarm Enable

| Value | Description |
|---|---|
| 0 | The UTC-matching alarm is disabled. |
| 1 | The UTC-matching alarm is enabled. |

### 26.6.10 RTC Status Register

**Name:** RTC_SR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TDERR | CALEV | TIMEV | SEC | ALARM | ACKUPD |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 5 – TDERR** Time and/or Date Free Running Error
If the RTC is configured in UTC mode, the value returned by this field is not relevant.

| Value | Name | Description |
|---|---|---|
| 0 | CORRECT | The internal free running counters are carrying valid values since the last read of the Status Register (RTC_SR). |
| 1 | ERR_TIMEDATE | The internal free running counters have been corrupted (invalid date or time, non-BCD values) since the last read and/or they are still invalid. |

**Bit 4 – CALEV** Calendar Event
The calendar event is selected in the CALEVSEL field in the Control Register (RTC_CR) and can be any one of the following events: week change, month change and year change. If the RTC is configured in UTC mode, the value returned by this field is not relevant.

| Value | Name | Description |
|---|---|---|
| 0 | NO_CALEVENT | No calendar event has occurred since the last clear. |
| 1 | CALEVENT | At least one calendar event has occurred since the last clear. |

**Bit 3 – TIMEV** Time Event
The time event is selected in the TIMEVSEL field in the Control Register (RTC_CR) and can be any one of the following events: minute change, hour change, noon, midnight (day change). If the RTC is configured in UTC mode, the value returned by this field is not relevant.

| Value | Name | Description |
|---|---|---|
| 0 | NO_TIMEVENT | No time event has occurred since the last clear. |
| 1 | TIMEVENT | At least one time event has occurred since the last clear. |

**Bit 2 – SEC** Second Event

| Value | Name | Description |
|---|---|---|
| 0 | NO_SECEVENT | No second event has occurred since the last clear. |
| 1 | SECEVENT | At least one second event has occurred since the last clear. |

**Bit 1 – ALARM** Alarm Flag

| Value | Name | Description |
|---|---|---|
| 0 | NO_ALARMEVENT | No alarm matching condition occurred. |
| 1 | ALARMEVENT | An alarm matching condition has occurred. |

**Bit 0 – ACKUPD** Acknowledge for Update

| Value | Name | Description |
|---|---|---|
| 0 | FREERUN | Time and calendar registers cannot be updated. |
| 1 | UPDATE | Time and calendar registers can be updated. |

### 26.6.11 RTC Status Clear Command Register

**Name:** RTC_SCCR
**Offset:** 0x1C
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Clears the corresponding status flag in the Status register (RTC_SR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TDERRCLR | CALCLR | TIMCLR | SECCLR | ALRCLR | ACKCLR |
| Access | | | W | W | W | W | W | W |
| Reset | | | – | – | – | – | – | – |

**Bit 5 – TDERRCLR** Time and/or Date Free Running Error Clear
If the RTC is configured in UTC mode, this bit has no effect.

**Bit 4 – CALCLR** Calendar Clear
If the RTC is configured in UTC mode, this bit has no effect.

**Bit 3 – TIMCLR** Time Clear
If the RTC is configured in UTC mode, this bit has no effect.

**Bit 2 – SECCLR** Second Clear

**Bit 1 – ALRCLR** Alarm Clear

**Bit 0 – ACKCLR** Acknowledge Clear

### 26.6.12 RTC Interrupt Enable Register

**Name:** RTC_IER
**Offset:** 0x20
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TDERREN | CALEN | TIMEN | SECEN | ALREN | ACKEN |
| Access | | | W | W | W | W | W | W |
| Reset | | | – | – | – | – | – | – |

**Bit 5 – TDERREN** Time and/or Date Error Interrupt Enable
If the RTC is configured in UTC mode, this bit has no effect.

**Bit 4 – CALEN** Calendar Event Interrupt Enable
If the RTC is configured in UTC mode, this bit has no effect.

**Bit 3 – TIMEN** Time Event Interrupt Enable
If the RTC is configured in UTC mode, this bit has no effect.

**Bit 2 – SECEN** Second Event Interrupt Enable

**Bit 1 – ALREN** Alarm Interrupt Enable

**Bit 0 – ACKEN** Acknowledge Update Interrupt Enable

### 26.6.13 RTC Interrupt Disable Register

**Name:** RTC_IDR
**Offset:** 0x24
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TDERRDIS | CALDIS | TIMDIS | SECDIS | ALRDIS | ACKDIS |
| Access | | | W | W | W | W | W | W |
| Reset | | | – | – | – | – | – | – |

**Bit 5 – TDERRDIS** Time and/or Date Error Interrupt Disable
If the RTC is configured in UTC mode, this bit has no effect.

**Bit 4 – CALDIS** Calendar Event Interrupt Disable
If the RTC is configured in UTC mode, this bit has no effect.

**Bit 3 – TIMDIS** Time Event Interrupt Disable
If the RTC is configured in UTC mode, this bit has no effect.

**Bit 2 – SECDIS** Second Event Interrupt Disable

**Bit 1 – ALRDIS** Alarm Interrupt Disable

**Bit 0 – ACKDIS** Acknowledge Update Interrupt Disable

### 26.6.14 RTC Interrupt Mask Register

**Name:** RTC_IMR
**Offset:** 0x28
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TDERR | CAL | TIM | SEC | ALR | ACK |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 5 – TDERR** Time and/or Date Error Mask
If the RTC is configured in UTC mode, this bit has no effect.

**Bit 4 – CAL** Calendar Event Interrupt Mask
If the RTC is configured in UTC mode, this bit is not relevant.

**Bit 3 – TIM** Time Event Interrupt Mask
If the RTC is configured in UTC mode, this bit is not relevant.

**Bit 2 – SEC** Second Event Interrupt Mask

**Bit 1 – ALR** Alarm Interrupt Mask

**Bit 0 – ACK** Acknowledge Update Interrupt Mask

### 26.6.15 RTC Valid Entry Register

**Name:** RTC_VER
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** Read-only

If the RTC is configured in UTC mode, the values returned by this register are not relevant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | NVCALALR | NVTIMALR | NVCAL | NVTIM |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 3 – NVCALALR** Non-valid Calendar Alarm

| Value | Description |
|---|---|
| 0 | No invalid data has been detected in RTC_CALALR (Calendar Alarm Register). |
| 1 | RTC_CALALR has contained invalid data since it was last programmed. |

**Bit 2 – NVTIMALR** Non-valid Time Alarm

| Value | Description |
|---|---|
| 0 | No invalid data has been detected in RTC_TIMALR (Time Alarm Register). |
| 1 | RTC_TIMALR has contained invalid data since it was last programmed. |

**Bit 1 – NVCAL** Non-valid Calendar

| Value | Description |
|---|---|
| 0 | No invalid data has been detected in RTC_CALR (Calendar Register). |
| 1 | RTC_CALR has contained invalid data since it was last programmed. |

**Bit 0 – NVTIM** Non-valid Time

| Value | Description |
|---|---|
| 0 | No invalid data has been detected in RTC_TIMR (Time Register). |

| Value | Description |
|---|---|
| 1 | RTC_TIMR has contained invalid data since it was last programmed. |

### 26.6.16 RTC TimeStamp Time Register 0

**Name:** RTC_TSTR0
**Offset:** 0xB0
**Reset:** 0x00000000
**Property:** Read-only

These fields are valid for non-UTC mode only.

RTC_TSTR0 reports the timestamp of the first tamper event after reading RTC_TSSR0.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BACKUP | | | | TEVCNT[3:0] | | | |
| Access | R | | | | R | R | R | R |
| Reset | 0 | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | AMPM | HOUR[5:0] | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | MIN[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | SEC[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – BACKUP** System Mode of the Tamper (cleared by reading RTC_TSSR0)

| Value | Description |
|---|---|
| 0 | The state of the system is different from Backup mode when the tamper event occurs. |
| 1 | The system is in Backup mode when the tamper event occurs. |

**Bits 27:24 – TEVCNT[3:0]** Tamper Events Counter (cleared by reading RTC_TSSR0)
Each time a tamper event occurs, this counter is incremented. This counter saturates at 15. Once this value is reached, it is no more possible to know the exact number of tamper events.

If this field is not null, this implies that at least one tamper event occurs since last register reset and that the values stored in timestamping registers are valid.

**Bit 22 – AMPM** AM/PM Indicator of the Tamper (cleared by reading RTC_TSSR0)

**Bits 21:16 – HOUR[5:0]** Hours of the Tamper (cleared by reading RTC_TSSR0)

**Bits 14:8 – MIN[6:0]** Minutes of the Tamper (cleared by reading RTC_TSSR0)

**Bits 6:0 – SEC[6:0]** Seconds of the Tamper (cleared by reading RTC_TSSR0)

### 26.6.17 RTC TimeStamp Time Register 0 (UTC_MODE)

**Name:** RTC_TSTR0 (UTC_MODE)
**Offset:** 0xB0
**Reset:** 0x00000000
**Property:** Read-only

RTC_TSTR0 reports the timestamp of the first tamper event after reading RTC_TSSR0.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BACKUP | | | | TEVCNT[3:0] | | | |
| Access | R | | | | R | R | R | R |
| Reset | 0 | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 31 – BACKUP** System Mode of the Tamper (cleared by reading RTC_TSSR0)

| Value | Description |
|---|---|
| 0 | The state of the system is different from Backup mode when the tamper event occurs. |
| 1 | The system is in Backup mode when the tamper event occurs. |

**Bits 27:24 – TEVCNT[3:0]** Tamper Events Counter (cleared by reading RTC_TSSR0)
Each time a tamper event occurs, this counter is incremented. This counter saturates at 15. Once this value is reached, it is no more possible to know the exact number of tamper events.

If this field is not null, this implies that at least one tamper event occurs since last register reset and that the values stored in timestamping registers are valid.

### 26.6.18 RTC TimeStamp Time Register 1

**Name:** RTC_TSTR1
**Offset:** 0xBC
**Reset:** 0x00000000
**Property:** Read-only

These fields are valid for non-UTC mode only.

RTC_TSTR1 reports the timestamp of the last tamper event after reading RTC_TSSR1.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BACKUP | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | AMPM | HOUR[5:0] | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | MIN[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | SEC[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – BACKUP** System Mode of the Tamper (cleared by reading RTC_TSSR1)

| Value | Description |
|---|---|
| 0 | The state of the system is different from Backup mode when the tamper event occurs. |
| 1 | The system is in Backup mode when the tamper event occurs. |

**Bit 22 – AMPM** AM/PM Indicator of the Tamper (cleared by reading RTC_TSSR1)

**Bits 21:16 – HOUR[5:0]** Hours of the Tamper (cleared by reading RTC_TSSR1)

**Bits 14:8 – MIN[6:0]** Minutes of the Tamper (cleared by reading RTC_TSSR1)

**Bits 6:0 – SEC[6:0]** Seconds of the Tamper (cleared by reading RTC_TSSR1)

### 26.6.19 RTC TimeStamp Time Register 1 (UTC_MODE)

**Name:** RTC_TSTR1 (UTC_MODE)
**Offset:** 0xBC
**Reset:** 0x00000000
**Property:** Read-only

RTC_TSTR1 reports the timestamp of the last tamper event after reading RTC_TSSR1.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BACKUP | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 31 – BACKUP** System Mode of the Tamper (cleared by reading RTC_TSSR1)

| Value | Description |
|---|---|
| 0 | The state of the system is different from Backup mode when the tamper event occurs. |
| 1 | The system is in Backup mode when the tamper event occurs. |

### 26.6.20 RTC TimeStamp Date Register

**Name:** RTC_TSDRx
**Offset:** 0xB4 + x*0x0C [x=0..1]
**Reset:** 0x00000000
**Property:** Read-only

These fields contain the date and the source of a tamper occurrence if RTC_TSTR0.TEVCNT is not null.

These fields are relevant for non-UTC mode only.

RTC_TSDR0 reports the timestamp of the first tamper event after reading RTC_TSSR0, and RTC_TSDR1 reports the timestamp of the last tamper event.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | DATE[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DAY[2:0] | | | MONTH[4:0] | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | YEAR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | CENT[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:24 – DATE[5:0]** Date of the Tamper (cleared by reading RTC_TSSRx)

**Bits 23:21 – DAY[2:0]** Day of the Tamper (cleared by reading RTC_TSSRx)

**Bits 20:16 – MONTH[4:0]** Month of the Tamper (cleared by reading RTC_TSSRx)

**Bits 15:8 – YEAR[7:0]** Year of the Tamper (cleared by reading RTC_TSSRx)

**Bits 6:0 – CENT[6:0]** Century of the Tamper (cleared by reading RTC_TSSRx)

### 26.6.21 RTC TimeStamp Date Register (UTC_MODE)

**Name:** RTC_TSDRx (UTC_MODE)
**Offset:** 0xB4
**Reset:** 0x00000000
**Property:** Read-only

RTC_TSDR0 reports the timestamp of the first tamper event after reading RTC_TSSR0, and RTC_TSDR1 reports the timestamp of the last tamper event.
This register is cleared by reading RTC_TSSRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | UTC_TIME[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UTC_TIME[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UTC_TIME[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UTC_TIME[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – UTC_TIME[31:0]** Time of the Tamper (UTC format)
This configuration is relevant only if UTC = 1 in RTC_MR.

### 26.6.22 RTC TimeStamp Source Register

**Name:** RTC_TSSRx
**Offset:** 0xB8 + x*0x0C [x=0..1]
**Reset:** 0x00000000
**Property:** Read-only

This register is cleared after read and the read access also performs a clear on RTC_TSTRx and RTC_TSDRx.
The following configuration values are valid for all listed bit names of this register:

0: No alarm generated since the last clear.

1: An alarm has been generated by the corresponding monitor since the last clear.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | JTAG | TST | | |
| Access | | | | | R | R | | |
| Reset | | | | | 0 | 0 | | |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – DETx** PIOBU Intrusion Detector (cleared on read)

**Bit 3 – JTAG** JTAG Pins Monitor (cleared on read)

**Bit 2 – TST** Test Pin Monitor (cleared on read)

# 27. System Controller Write Protection (SYSCWP)

## 27.1 Functional Description

### 27.1.1 System Controller Peripherals Mapping

**Table 27-1. System Controller Peripherals Mapping**

| Offset | System Controller Peripheral | Name |
|---|---|---|
| 0x000-0x00C | Reset Controller | RSTC |
| 0x010-0x01C | Shutdown Controller | SHDWC |
| 0x030-0x03C | Period Interval Timer | PIT |
| 0x040-0x04C | Watchdog Timer | WDT |
| 0x050-0x05C | Slow Clock Controller | SCKC |
| 0x0B0-0x190 | Real Time Clock | RTC |
| 0x194 | Write Protection Mode Register | SYSC_WPMR |

### 27.1.2 Register Write Protection

To prevent any single software error from modifying the configuration of the Reset Controller (RSTC), Shutdown Controller (SHDWC), Periodic Interval Timer (PIT), Slow Clock Controller (SCKC), Real-time Clock (RTC) and Watchdog Timer (WDT), some registers of these peripherals can be write-protected by setting the WPEN bit in the System Controller Write Protection Mode register (SYSC_WPMR).

**Note:** The WDT embeds additional write protection mechanisms.

The following registers can be write-protected when SYSC_WPMR.WPEN=1:

- WDT Control Register
- WDT Mode Register
- RSTC Mode Register
- SHDWC Mode Register
- SHDWC Wakeup Inputs Register
- PIT Mode Register
- SCKC Configuration Register
- RTC Control Register
- RTC Mode Register
- RTC Time Alarm Register
- RTC Calendar Alarm Register

## 27.2 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | SYSC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |

### 27.2.1 SYSC Write Protection Mode Register

**Name:** SYSC_WPMR
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x535943 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable

| Value | Description |
|---|---|
| 0 | Disables the write protection of the configuration registers if WPKEY corresponds to 0x535943 (“SYC” in ASCII). |
| 1 | Enables the write protection of the configuration registers if WPKEY corresponds to 0x535943 (“SYC” in ASCII). |

# 28. Slow Clock Controller (SCKC)

## 28.1 Description

The System Controller embeds a Slow Clock Controller (SCKC). The SCKC selects the slow clock from one of two sources:

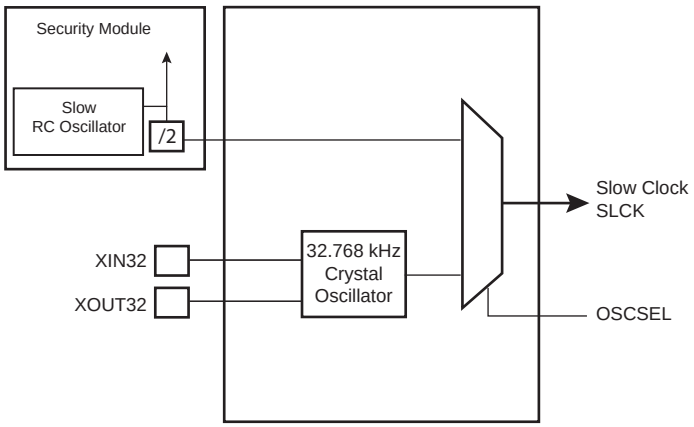
- External 32.768 kHz crystal oscillator
- Embedded 64 kHz (typical) RC oscillator

## 28.2 Embedded Characteristics

- 64 kHz (typical) RC Oscillator or 32.768 kHz Crystal Oscillator Selector
- VDDBU Powered

## 28.3 Block Diagram

**Figure 28-1. Block Diagram**




## 28.4 Functional Description

The OSCSEL bit is located in the Slow Clock Controller Configuration register (SCKC_CR), in the backup domain, and its value is kept while VDDBU is present.

The embedded 64 kHz (typical) Slow RC oscillator and the 32.768 kHz crystal oscillator are always enabled as soon as VDDBU is established. The Slow Clock Selector command (OSCSEL bit) selects the slow clock source.

After the VDDBU power-on reset, the default configuration is OSCSEL = 0, allowing the system to start on the embedded 64 kHz (typical) RC oscillator.

The programmer controls the slow clock switching by software, so precautions must be taken during the switching phase.

### 28.4.1 Switching from Embedded 64 kHz RC Oscillator to 32.768 kHz Crystal Oscillator

The sequence to switch from the embedded 64 kHz (typical) RC oscillator to the 32.768 kHz crystal oscillator is the following:

1. Switch the master clock to a source different from slow clock (PLL or Main Oscillator) through the Power Management Controller.
2. Switch from the embedded 64 kHz (typical) RC oscillator to the 32.768 kHz crystal oscillator by writing a 1 to the OSCSEL bit.
3. Wait 5 slow clock cycles for internal resynchronization.

### 28.4.2 Switching from 32.768 kHz Crystal Oscillator to Embedded 64 kHz RC Oscillator

The sequence to switch from the 32.768 kHz crystal oscillator to the embedded 64 kHz (typical) RC oscillator is the following:

1. Switch the master clock to a source different from slow clock (PLL or Main Oscillator).
2. Switch from the 32.768 kHz crystal oscillator to the embedded RC oscillator by writing a 0 to the OSCSEL bit.
3. Wait 5 slow clock cycles for internal resynchronization.

## 28.5 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | SCKC_CR | 7:0 | | | | | OSCSEL | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 28.5.1 Slow Clock Controller Configuration Register

**Name:** SCKC_CR
**Offset:** 0x0
**Reset:** 0x00000001
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode register (SYSC_WPMR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSCSEL | | | |
| Access | | | | | R/W | | | |
| Reset | | | | | 0 | | | |

**Bit 3 – OSCSEL** Slow Clock Selector

| Value | Description |
|---|---|
| 0 (RC) | Slow clock is the embedded 64 kHz (typical) RC oscillator. |
| 1 (XTAL) | Slow clock is the 32.768 kHz crystal oscillator. |

# 29. Peripheral Touch Controller (PTC)

## 29.1 Description

The QTouch Peripheral Touch Controller (PTC) subsystem offers built-in hardware for capacitive touch measurement on sensors that function as buttons, sliders and wheels. The PTC subsystem supports both mutual and self-capacitance measurement without the need for any external component. It offers sensitivity and noise tolerance, as well as self-calibration, and minimizes the sensitivity tuning effort by the user.

The PTC subsystem is intended for autonomously performing capacitive touch sensor measurements. The external capacitive touch sensor is typically formed on a PCB, and the sensor electrodes are connected to the analog charge integrator of the PTC using the device I/O pins. The PTC supports mutual capacitance sensors organized as capacitive touch matrices in different X-Y configurations. In Mutual Capacitance mode, the PTC requires one pin per X line (drive line) and one pin per Y line (sense line). In Self-capacitance mode, the PTC requires only one pin with a Y-line driver for each self-capacitance sensor.

## 29.2 Embedded Characteristics

- Implements Low-power, High-sensitivity, Environmentally-robust Capacitive Touch Buttons, Sliders, and Wheels
  - One Pin per Electrode – No External Components
  - Zero Drift over Temperature and supply/reference ranges
  - No Need for Temperature or supply/reference compensation
- “On demand” or “Timed” measurement
- Supports Mutual Capacitance and Self-capacitance Sensing
  - Up to 8 Buttons in Self-capacitance Mode
  - Up to 64 Buttons in Mutual Capacitance Mode
  - Supports Lumped Mode Configuration(1)
- Calibration
  - Load Compensating Charge Sensing
  - Parasitic capacitance compensation together with the electrode capacitance
- Adjustable Gain for Higher Sensitivity
  - Analog Gain 1 to 16
  - Digital Gain 1 to 32
- Noise Immunity
  - Hardware Noise Filtering by Accumulation 1 to 64
  - Adjacent Key Suppression (AKS), Removal of False Detection(2)
  - Frequency Hopping: Noise Signal Desynchronization for High Conducted Immunity(3)
- Provided PTC Subsystem Firmware(4)
- Acquisition Module (Node Definitions, pPP and PTC Management) is Product-dependent, which implements all Hardware-dependent Operations for Configuration and Measurement of Capacitive Touch or Proximity Sensors

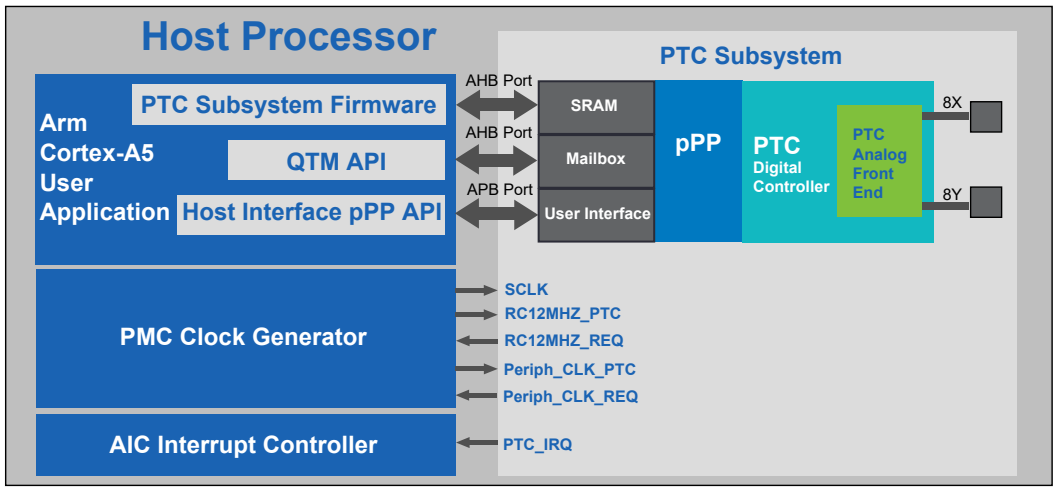
- Signal Conditioning Module (Frequency Hopping) applies Algorithmic and Feedback Control Methods to improve the Quality of Measurement Data captured by an Acquisition Module
- Post-processing Modules (Key, Scroller) interpret Measurement Data in the Context of a Capacitive Touch or Proximity Sensor
- Scroller Module defines Slider and Wheels Configuration and Data, based on Keys Module Setting

**Note:**

1. A lumped sensor is implemented as a combination of multiple sense lines (self-capacitance measurement) or multiple drive and sense lines (mutual capacitance measurement) to act as one single button sensor. This provides the application developer with greater flexibility in the touch sensor implementation.
2. The PTC incorporates the Adjacent Key Suppression (AKS) technology, which can be selected on a per-key basis. The AKS technology is used to suppress multiple key presses based on relative signal strength. This feature assists in solving the problem of surface moisture which can bridge a key touch to an adjacent key, causing multiple key presses.
3. This PTC subsystem supports frequency hopping, which tries to select a sampling frequency that does not clash with noise at specific frequencies elsewhere in products or product operating environments. Frequency Hopping tries to hop away from the noise.
4. It is necessary to use the firmware provided by Microchip in order to use the PTC subsystem.

## 29.3 Block Diagram

**Figure 29-1. PTC Block Diagram**




**Note:** QTM is the QTouch Manager firmware interface.

## 29.4 Signal Description

**Table 29-1. Signal Description**

| Name | Type | Description |
|---|---|---|
| PTC_X[n..0] | Output | 8 X-lines with n=7<br>Transmit lines in Mutual Capacitance mode. |

| .....continued | | |
|---|---|---|
| **Name** | **Type** | **Description** |
| PTC_Y[m..0] | Input | 8 Y-lines with m=7<br>Receive lines in Mutual/Self-capacitance mode. |
| SCLK | Input | Connection to the product system 32 kHz slow clock |
| RC12MHZ_PTC | Input | Direct connection to the 12 MHz RC oscillator |
| RC12MHZ_REQ | Output | Request to supply the RC12MHz PTC subsystem clock |
| Periph_CLK_PTC | Input | Peripheral clock enabled by the PTC ID (max 83 MHz) |
| Periph_CLK_REQ | Output | Request to supply the peripheral clock |
| PTC_IRQ | Output | One PTC IRQ rises for host flag (28, 29, 30 or 31). |

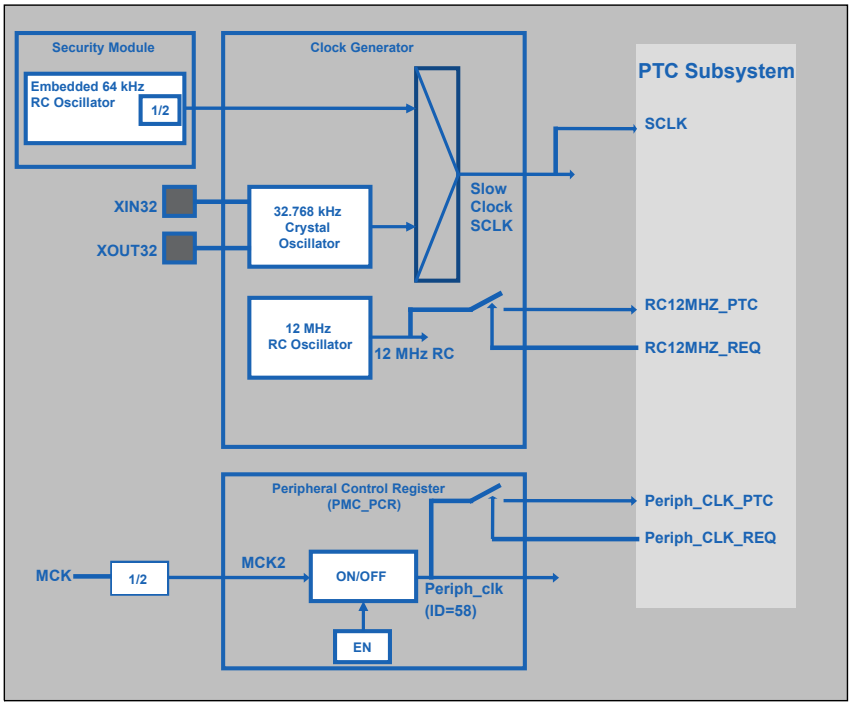
## 29.5 Product Dependencies

The PTC subsystem needs to have some other peripherals of the Arm system configured correctly, as described in the following sections. Those peripherals are the PIO Controller (PIO), the Advanced Interrupt Controller (AIC) and the Power Management Controller (PMC).

### 29.5.1 Power Management

The PTC Controller is not continuously clocked. The programmer must first enable the PTC Controller peripheral clock in the Power Management Controller (PMC) before using the PTC Controller. However, if the application does not require PTC operations, the PTC Controller clock can be stopped when not needed and restarted when necessary. Configuring the PTC Controller requires the PTC Controller clock to be enabled.

**Figure 29-2. PTC Subsystem Clock Sources**



The PTC subsystem operates both from a peripheral clock synchronous to the master clock of the system and from an asynchronous clock source directly connected to the embedded 12 MHz RC oscillator. The selected clocks must be enabled in the PMC before they can be used by the PTC. By default, the 12 MHz RC oscillator is enabled at startup of the product.

The various clock sources are as follows:

- PERIPH_CLK_PTC

This clock source is dedicated to the picoPower processor. It is located in the PMC as Periph_clk[PID]=PCLOCK_LS. This clock is synchronous with the AHB/APB matrix controlling the host interface and the mailbox. The clock frequency is between 12 MHz and 83 MHz. The same clock is used for the Arm interface connected as an APB slave via an AHB/APB bridge. It is also used to program the code/data SRAM and to access the mailbox SRAM.

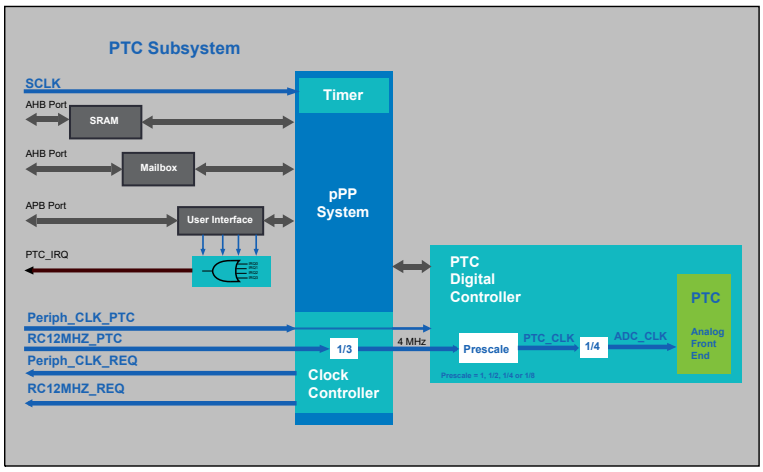
- RC12MHZ

A different clock is used for the PTC digital controller. This clock can be divided internally in the pPP before being used. There is also a small local prescaler in the PTC digital controller to allow lower clock rates. Thus, the PTC operates from an asynchronous clock source and the operation is independent from the main system clock and its derivative clocks, such as the peripheral bus clock (PERIPH_CLK_PTC).

- SCLK

For the timers, a 32 kHz clock is used and divided internally down to a 1 kHz clock for counting the timer interrupt.

**Figure 29-3. PTC Subsystem Clock Schematic**




The RC12HMZ clock is internally divided by 3 in the PTC subsystem, and so a 4 MHz clock is provided to the PTC digital controller. This controller can divide the clock further by 1, 2, 4 or 8 to slow down the PTC clock.

- ADC_CLK

The prescaled clock PTC_CLK is divided by 4 to supply an ADC_CLK to the PTC analog front end.

The ADC data rate is defined by the controller. The typical value is about 33 kHz to 66 kHz depending on the timing configuration.

### 29.5.2 I/O Lines

The pins used for interfacing the PTC may be multiplexed with GPIO lines. When the PTC subsystem is activated and the X-Y lines selected, the GPIO switches automatically to analog state. The ADC modules

possibly hanging on the same PTC analog lines should not use the same GPIO for ADC conversion. Adjacent GPIO lines to PTC lines must not be used to output high speed signals to avoid crosstalk with PTC sensing.

When a line is disabled, the corresponding I/O pin is not reserved for the PTC subsystem, and it can be used for some alternative I/O function.

There is an individual selection bit for each Y or X line. In normal cases, just one line should be active at the same time. For more advanced uses, like proximity sensing, several lines may be selected in parallel. The input and output functionality, such as charging and sensing pulses of the selected line, is controlled automatically by the PTC digital controller sequencer in various operating modes. The I/O lines used for analog PTC_X lines and PTC_Y lines must be connected to external capacitive touch sensor electrodes. External components are not required for normal operation.

### 29.5.3 Interrupt Sources

The PTC_IRQ interrupt line is connected on one of the internal sources of the host processor interrupt controller (AIC). Using the PTC_IRQ interrupt requires the interrupt controller to be programmed first.

Four interrupts (IRQ0,1,2,3) can be generated in the host interface register. The PTC_IRQ line is a logical “OR” between the four IRQ0, IRQ1, IRQ2 and IRQ3.

## 29.6 Functional Description

The PTC analog front end (AFE) and the digital controller are not managed directly by the Cortex-A5 processor. An intermediate processor (pPP) is introduced to manage all functionalities of the PTC. A pPP program, a “firmware”, is needed. This program is loaded by the Arm (host) in a shared SRAM area. The firmware embeds many software functionalities and algorithms to ensure an efficient touch detection.

### 29.6.1 picoPower Processor (pPP)

The picoPower Processor (pPP) is a small processor dedicated to handling the PTC and to processing its data in order to offload the main host Arm processor.

The pPP uses a unified memory architecture where instructions and data share the address space. The pPP embeds a 16 Kbytes SRAM block. When the processor is stopped, the 16 Kbytes SRAM block can be used by the Arm processor.

The pPP has single-cycle access to instructions and data that reside in the SRAM.

Loads and stores go to the local code/data SRAM, the shared mailbox SRAM or the local I/O space.

Accesses to the mailbox enter a wait state for every other access cycle.

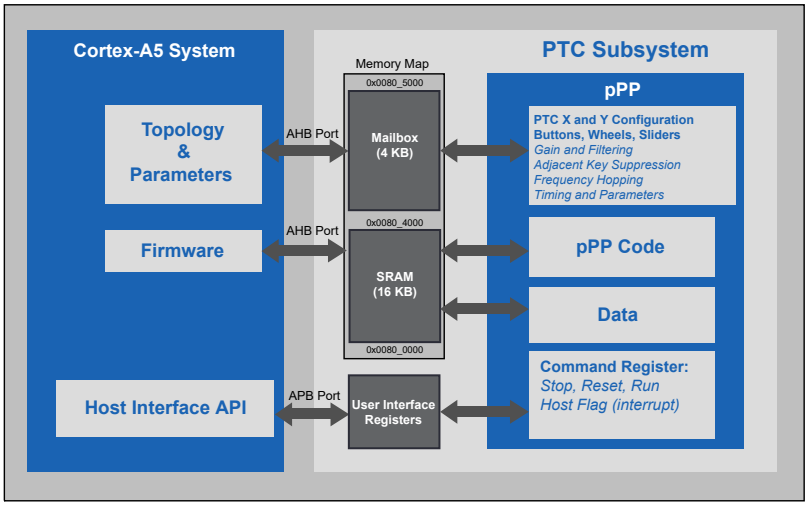
Accesses to code/data SRAM and local I/O space never enter wait states.

### 29.6.2 Shared Memories

The SRAM memory space contains context save, interrupt vectors and a unified instruction/data space that can be used for stacks and instructions.

On top of the SRAM space is the shared mailbox SRAM.

**Figure 29-4. Memory Map**




#### 29.6.2.1 Mailbox

The mailbox (4 Kbytes) is used to indicate to the pPP the X and Y topoloy as well as the number of sensors implemented.

The mailbox can also be used, for example, to pass the parameters required by an application to adjust the analog or digital gain, the filtering functions and some touch operation timing values.

The mailbox is the main way to control the PTC digital and analog components. The pPP firmware reads the mailbox and performs the tasks requested. After execution, some data are fed back to the mailbox to be read by the main processor, such as touch button confirmation or touch position on the slider or wheel.

#### 29.6.2.2 SRAM Data Area

The pPP uses the SRAM data area for its own needs and to work with the firmware local variables. This SRAM section is not used to communicate with the Arm.

#### 29.6.2.3 Firmware in SRAM Code Area

The firmware contains all PTC subsystem functionalities, allowing PTC measurement in the different conditions of parameters and configurations. The firmware is a binary file copied to the SRAM code area at the address defined by the memory map. The firmware makes the pPP work properly with some peripherals like the timer, a clock generator and obviously the PTC digital controller and the PTC analog front end. The firmware embeds all QTML (QTouch Modular Library) functionalities. Those modules are not modifiable by the application developer. The QTML functionalities configuration and data are controllable by the mailbox. The host has read and write accesses to the mailbox.

#### 29.6.2.4 Host Interface

The pPP can be controlled by the host processor through an APB interface and the user interface registers. This is referred to as the “host interface”. Some configurations are only accessible when the pPP is stopped. The host interface includes pPP flags, which are also called host flags on the firmware, for interprocessor communications. The user interface registers can run, stop and reset the pPP and read the IRQ host flags. Nevertheless, the mailbox remains the main means of communication.

##### 29.6.2.4.1 Processor Command Registers

The CMD field is part of the host interface and is used to start, stop and reset the pPP. Writing a valid command to this field changes the internal state of the pPP. After a number of cycles, this state change is reflected in the processor state register.

When a START command is issued, the host is no longer able to write to host interface registers which are marked as run-time write-locked. The pPP RAM block is also locked by this command. The host interface registers and RAM block can be unlocked by using the STOP or RESET commands. The lock is released when the processor state register reflects this state.

### 29.6.3 PTC Digital Controller

The PTC digital controller is a peripheral of the pPP. It is intended for acquiring capacitive touch sensor and capacitive proximity sensor signals under limited firmware control by the controlling processor. The PTC digital controller consists of an Analog Charge Integrator and a 10-bit ADC Controller, 16-bit Digital Accumulator for the ADC results and a State Machine taking care of sensor sampling and digital accumulation sequence.

#### 29.6.3.1 PTC Digital Controller Operations

- Sensing mode (mutual or self)
- Control of the ADC 10-bit SAR state machine single ADC conversion or free run mode (comparator and ADC data/accumulator register)
- Digital gain up to 32 and averaging up to 64 ADC codes
- Selection of the filtering resistance (0, 20, 50 or 100 kΩ)
- Adjustment of the compensation capacitor up to 30 pF
- Adjustment of the integration capacitor up to 30 pF
- Frequency hopping(1) implementation (modification of the sampling rate to avoid synchronous parasitic noise)
- Channel Change Delay Selection CDS(2) (settling time)
- Prescaling (1, 1/2, 1/4, 1/8), 4 MHz down to ADC_CLK

**Note:**

1. A programmable sampling delay can be used to choose (modify) the sampling frequency that is best suited in an application where other periodic noise sources may otherwise disturb the sampling. Frequency hopping can also be modified automatically from one sampling cycle to another, by setting the software driver parameters.
2. CDS bits define the delay when changing input channels. The delay allows the analog circuits to settle on a new (Y) channel or channel pair (X-Y). The delay is application-dependent, and therefore this option enables the user to select a suitable delay. The delay is expressed as a number of PTC clock cycles.

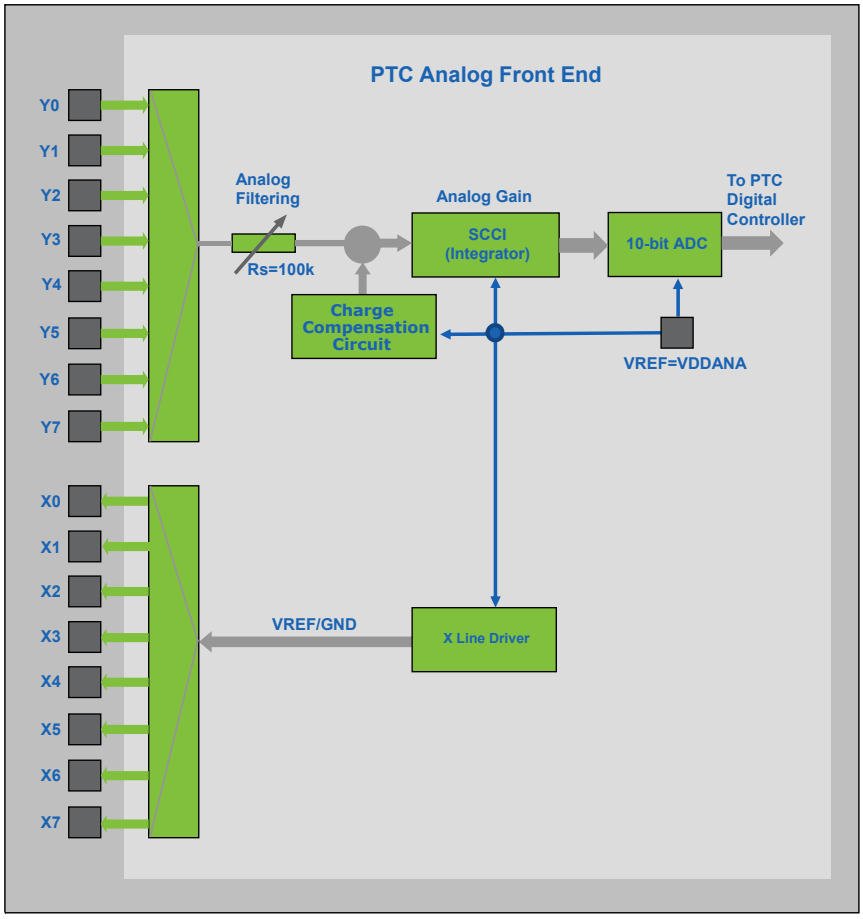
### 29.6.4 PTC Analog Front End (AFE)

The analog front end consists of X-line drivers, a sensor capacitance compensation circuit and a parasitic capacitance insensitive analog Switched Capacitor Charge Integrator (SCCI). The integrator is connected to sensor Y-lines via an analog multiplexer. When the PTC digital controller is enabled, the SCCI output is automatically connected to the ADC input.

The external capacitive touch sensor is typically formed on a PCB and the sensor electrodes are connected to the Analog Charge Integrator of the PTC AFE via MCU I/O port pins. The PTC AFE supports mutual capacitance sensors organized as capacitive touch matrices in different X-Y configurations (QTouch Surface). The PTC AFE requires one pin per X-line and one pin per Y-line. No external components are needed.

The PTC AFE also supports “self-capacitance touch sensors” (QTouch). In Self-capacitance mode, the PTC AFE requires just one Y-line pin per self-capacitance sensor.

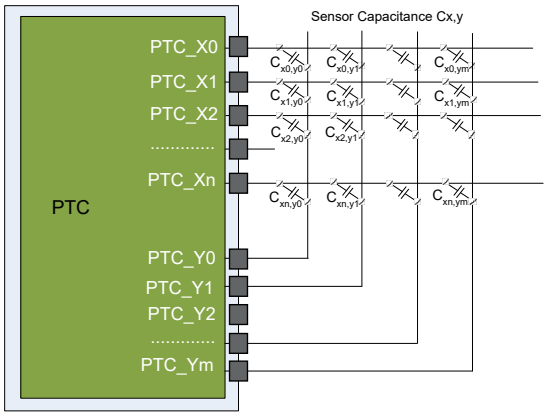
**Figure 29-5. PTC Analog Front End**




### 29.6.5 Operations in Mutual Capacitance

A mutual capacitance sensor is formed between two I/O lines, a PTC_X electrode for transmitting, and a PTC_Y electrode for receiving. The mutual capacitance between the PTC_X and PTC_Y electrodes is calibrated and measured by the PTC. It is not necessary to connect all X and Y lines; when unused, they can be left unconnected.

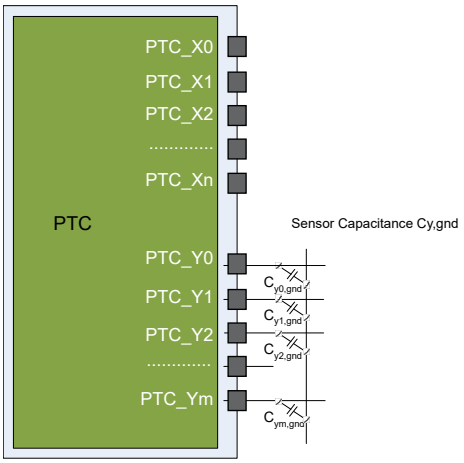
**Figure 29-6. Mutual Capacitance Sensor Arrangement**



### 29.6.6 Operations in Self-capacitance

The self-capacitance sensor is connected to a single pin on the PTC through the PTC_Ym electrodes to receive the signal. The sensor electrode capacitance is measured by the PTC.

**Figure 29-7. Self-capacitance Sensor Arrangement**



## 29.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x28 | PTC_CMD | 7:0 | | | | | CMD[3:0] | | | |
| 0x29<br>...<br>0x2F | Reserved | | | | | | | | | |
| 0x30 | PTC_ISR | 7:0 | IRQ3 | IRQ2 | IRQ1 | IRQ0 | | | | NOTIFY0 |
| 0x31<br>...<br>0x34 | Reserved | | | | | | | | | |
| 0x35 | PTC_IED | 7:0 | IER3 | IER2 | IER1 | IER0 | | | | |

### 29.7.1 PTC Command Register

**Name:** PTC_CMD
**Offset:** 0x28
**Reset:** –
**Property:** Write-only

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CMD[3:0] | | | |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

**Bits 3:0 – CMD[3:0]** Host Command
Issues commands to the pPP.

| Value | Name | Description |
|---|---|---|
| 0x0 | NO_ACTION | – |
| 0x1 | STOP | Waits for ongoing execution to complete, then stops. |
| 0x2 | RESET | Stops and resets. |
| 0x3 | Reserved | – |
| 0x4 | ABORT | Stops without waiting for ongoing execution to complete. |
| 0x5 | RUN | Starts execution (from stopped state). |
| 0x6–0xF | Reserved | – |

### 29.7.2 PTC Interrupt Status Register

**Name:** PTC_ISR
**Offset:** 0x30
**Reset:** 0x00
**Property:** Read/Write

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IRQ3 | IRQ2 | IRQ1 | IRQ0 | | | | NOTIFY0 |
| Access | R/W | R/W | R/W | R/W | | | | R/W |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

**Bits 4, 5, 6, 7 – IRQx** Interrupt to the Host
Used for communications between the host processor and the pPP. The firmware can set an IRQ event in fields IRQ0 to IRQ3. Any of the pPP IRQ0 to IRQ3 fields automatically rises the PTC_IRQ signal.

**Bit 0 – NOTIFY0** Notification to the Firmware
Used for communications between the host processor and the pPP. The firmware is notified when a command is used.

### 29.7.3 PTC Enable Register

**Name:** PTC_IED
**Offset:** 0x35
**Reset:** –
**Property:** Write-only

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IER3 | IER2 | IER1 | IER0 | | | | |
| Access | W | W | W | W | | | | |
| Reset | – | – | – | – | | | | |

**Bits 4, 5, 6, 7 – IERx** Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables interrupt for device-to-host interrupt. |

# 30. Low Power Asynchronous Receiver (RXLP)

## 30.1 Description

The Low Power Asynchronous Receiver (RXLP) is a low-power UART with a slow clock. It works only in Receive mode. It features a Receive Data (RXD) pin that can be used to wake up the system. The wake-up occurs only on data matching. Expected data can be a single value, two values, or a range of values.

The RXLP operates on a slow clock domain to reduce power consumption.

## 30.2 Embedded Characteristics

- Exit from Backup Mode on Comparison Match
- Programmable Baud Rate Generator
- Even, Odd, Mark or Space Parity Check
- Parity and Framing Error Detection
- Digital Filter on Receive Line
- Comparison Function on Received Character
- Register Write Protection

## 30.3 Block Diagram

**Figure 30-1. Block Diagram**

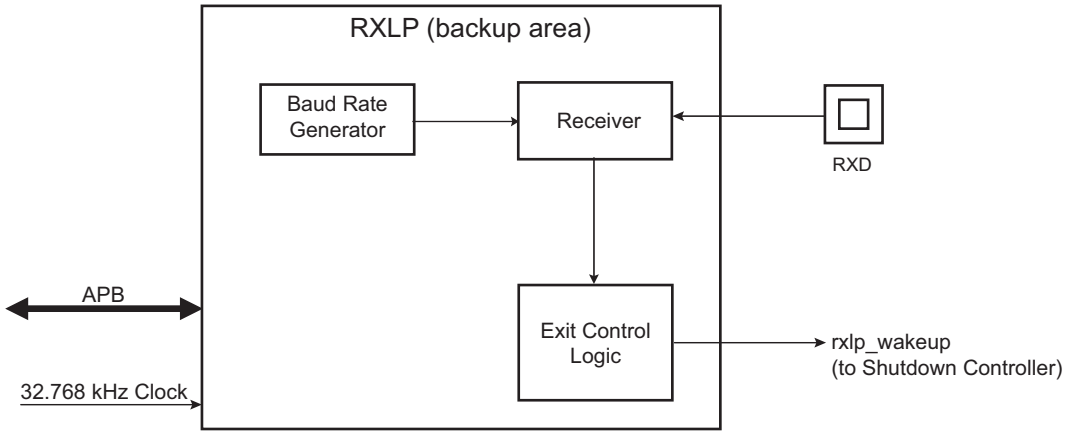



**Table 30-1. Pin Description**

| Pin Name | Description | Type |
|---|---|---|
| RXD | RXLP Receive Data | Input |

## 30.4 Product Dependencies

### 30.4.1 Power Management

The peripheral clock is not managed by the PMC. It is automatically activated when the RXLP is enabled and receive line is active. The peripheral clock is automatically de-activated after transmission of the wakeup signal.

## 30.5 Functional Description

The RXLP features an RS232 receive-only circuitry able to decode and compare data and parity while the system is in Backup mode. If a matching comparison occurs, the RXLP instructs the system to wake up (if enabled).

The RXLP operates in Asynchronous mode only and supports only 8-bit character handling (with or without parity).

The RXLP is made up of a receiver and a baud rate generator. Receiver timeout is not implemented and there is no interrupt line.

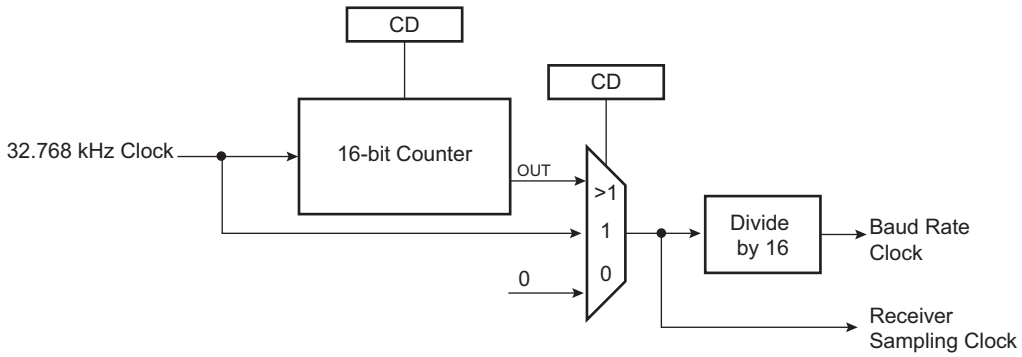
### 30.5.1 Baud Rate Generator

The baud rate generator provides the bit period clock named baud rate clock to the receiver.

The baud rate clock is the 32.768 kHz clock from the crystal oscillator, divided by 16 times the value (CD) written in the Baud Rate Generator register (RXLP_BRGR). If RXLP_BRGR is set to 0, the baud rate clock is disabled and the RXLP remains inactive. The maximum allowable baud rate is 32.768 kHz clock divided by 16. The minimum allowable baud rate is 32.768 kHz clock divided by (16 × 3).

$$\text{Baud Rate} = \frac{f_{32.768\ \text{kHz clock}}}{16 \times \text{CD}}$$

**Figure 30-2. Baud Rate Generator**




### 30.5.2 Receiver

#### 30.5.2.1 Receiver Reset, Enable and Disable

After device reset, the RXLP is disabled and must be enabled before being used. The receiver can be enabled by setting bit RXEN in the Control register (RXLP_CR). At this command, the receiver starts looking for a start bit.

The programmer can disable the receiver by setting bit RXLP_CR.RXDIS. If the receiver is waiting for a start bit, it is immediately stopped. However, if the receiver has already detected a start bit and is receiving the data, it waits for the stop bit before actually stopping its operation.

The receiver can be put in reset state by setting bit RXLP_CR.RSTRX. In this case, the receiver immediately stops its current operations and is disabled, whatever its current state. If RSTRX is applied when data is being processed, this data is lost. After initiating a reset it is mandatory to clear bit RXLP_CR.RSTRX.

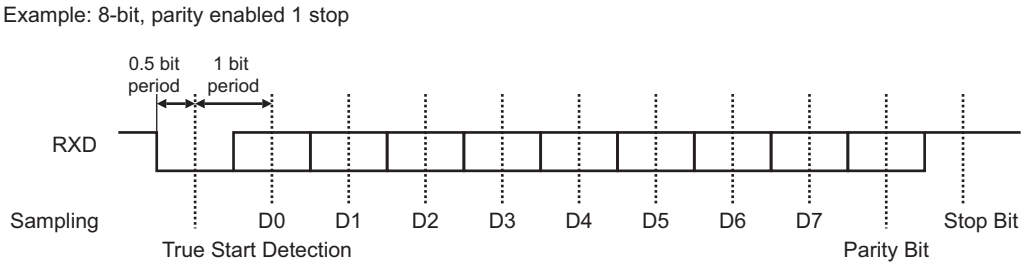
#### 30.5.2.2 Start Detection and Data Sampling

The RXLP only supports asynchronous operations, and this affects only its receiver. The RXLP detects the start of a received character by sampling the RXD signal until it detects a valid start bit. A low level (space) on RXD is interpreted as a valid start bit if it is detected for more than seven cycles of the sampling clock, which is 16 times the baud rate. Hence, a space that is longer than 7/16 of the bit period is detected as a valid start bit. A space which is 7/16 of a bit period or shorter is ignored and the receiver continues to wait for a valid start bit.

When a valid start bit has been detected, the receiver samples the RXD at the theoretical midpoint of each bit. It is assumed that each bit lasts 16 cycles of the sampling clock (1-bit period) so the bit sampling point is eight cycles (0.5-bit period) after the start of the bit. The first sampling point is therefore 24 cycles (1.5-bit periods) after detecting the falling edge of the start bit.

Each subsequent bit is sampled 16 cycles (1-bit period) after the previous one.

**Figure 30-3. Character Reception**




#### 30.5.2.3 Parity Error

Each time a character is received, the receiver calculates the parity of the received data bits, in accordance with the field PAR in the Mode register (RXLP_MR). It then compares the result with the received parity bit. If different, the received character is ignored and the receiver continues to wait for a new valid start bit.

#### 30.5.2.4 Receiver Framing Error

When a start bit is detected, it generates a character reception when all the data bits have been sampled. The stop bit is also sampled and when it is detected at 0, the received character is ignored and the receiver continues to wait for a new valid start bit.

#### 30.5.2.5 Receiver Digital Filter

The RXLP embeds a digital filter on the receive line. It is disabled by default and can be enabled by writing a logical 1 in RXLP_MR.FILTER. When enabled, the receive line is sampled using the 16x bit clock and a three-sample filter (majority 2 over 3) determines the value of the line.

### 30.5.3 Comparison Function on Received Character

Each time a valid character is received (without parity error and without frame error) it is compared to the wake-up trigger values. If the received character matches to the condition of wake-up, it is stored in the Receiver Holding register (RXLP_RHR), a system wake-up is generated and the RXLP is automatically disabled. If the character received does not match, it is ignored and the receiver continues to wait for a new valid start bit.

RXLP_CMPR (see RXLP Comparison Register) can be programmed to provide three different comparison methods:

- VAL1 equals VAL2—the comparison is performed on a single value and the wake-up request is generated if the received character equals VAL1.
- VAL1 is strictly lower than VAL2—any value between VAL1 and VAL2 generates a wake-up request.
- VAL1 is strictly higher than VAL2—the wake-up request is generated if either received character equals VAL1 or VAL2.

### 30.5.4 Register Write Protection

To prevent any single software error from corrupting RXLP behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the RXLP Write Protection Mode Register (RXLP_WPMR).

The following registers can be write-protected:

- RXLP Mode Register
- RXLP Baud Rate Generator Register
- RXLP Comparison Register

## 30.6 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | RXLP_CR | 7:0 | | | RXDIS | RXEN | | RSTRX | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | RXLP_MR | 7:0 | | | | FILTER | | | | |
| | | 15:8 | | | | | PAR[2:0] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 ... 0x17 | Reserved | | | | | | | | | |
| 0x18 | RXLP_RHR | 7:0 | RXCHR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C ... 0x1F | Reserved | | | | | | | | | |
| 0x20 | RXLP_BRGR | 7:0 | | | | | | | CD[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 | RXLP_CMPR | 7:0 | VAL1[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | VAL2[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x28 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | RXLP_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |

### 30.6.1 RXLP Control Register

**Name:** RXLP_CR
**Offset:** 0x0000
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | RXDIS | RXEN | | RSTRX | | |
| Access | | | W | W | | W | | |
| Reset | | | – | – | | – | | |

**Bit 5 – RXDIS** Receiver Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The receiver is disabled. If a character is being processed and RSTRX is not set, the character is completed before the receiver is stopped. |

**Bit 4 – RXEN** Receiver Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The receiver is enabled if RXDIS is 0. |

**Bit 2 – RSTRX** Reset Receiver

| Value | Description |
|---|---|
| 0 | Deactivates the reset of the receiver logic. |
| 1 | The receiver logic is reset and disabled. If a character is being received, the reception is aborted. The receiver logic remains in reset state until RSTRX is written to 0. |

### 30.6.2 RXLP Mode Register

**Name:** RXLP_MR
**Offset:** 0x0004
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PAR[2:0] | | | |
| Access | | | | | R/W | R/W | R/W | |
| Reset | | | | | 0 | 0 | 0 | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | FILTER | | | | |
| Access | | | | R/W | | | | |
| Reset | | | | 0 | | | | |

**Bits 11:9 – PAR[2:0]** Parity Type

| Value | Name | Description |
|---|---|---|
| 0 | EVEN | Even parity |
| 1 | ODD | Odd parity |
| 2 | SPACE | Parity forced to 0 |
| 3 | MARK | Parity forced to 1 |
| 4 | NO | No parity |

**Bit 4 – FILTER** Receiver Digital Filter

| Value | Name | Description |
|---|---|---|
| 0 | DISABLED | RXLP does not filter the receive line. |
| 1 | ENABLED | RXLP filters the receive line using a three-sample filter (16x-bit clock) (2 over 3 majority). |

### 30.6.3 RXLP Receiver Holding Register

**Name:** RXLP_RHR
**Offset:** 0x0018
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXCHR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – RXCHR[7:0]** Received Character
Last received character

### 30.6.4 RXLP Baud Rate Generator Register

**Name:** RXLP_BRGR
**Offset:** 0x0020
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CD[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bits 1:0 – CD[1:0]** Clock Divisor

| Value | Description |
|---|---|
| 0 | Baud rate clock is disabled |
| 1 | $f_{32.768\ kHz\ clock}$ / (CD × 16) |
| 2 | $f_{32.768\ kHz\ clock}$ / (CD × 16) |
| 3 | $f_{32.768\ kHz\ clock}$ / (CD × 16) |

### 30.6.5 RXLP Comparison Register

**Name:** RXLP_CMPR
**Offset:** 0x0024
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | VAL2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | VAL1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – VAL2[7:0]** Second Comparison Value for Received Character
0 to 255.

The received character must be lower or equal to the value of VAL2 and higher or equal to VAL1 to request a system wake-up.

**Bits 7:0 – VAL1[7:0]** First Comparison Value for Received Character
0 to 255.

The received character must be higher or equal to the value of VAL1 and lower or equal to VAL2 to request a system wake-up.

### 30.6.6 RXLP Write Protection Mode Register

**Name:** RXLP_WPMR
**Offset:** 0x00E4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x525840 | PASSWD | Writing any other value in this field aborts the write operation. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See Register Write Protection for the list of registers that can be protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x52584C (RXL in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x52584C (RXL in ASCII). |

# 31. Clock Generator

## 31.1 Description

The Clock Generator User Interface is embedded within the Power Management Controller and is described in 32.22 Register Summary. However, the Clock Generator registers are named CKGR_.

## 31.2 Embedded Characteristics

The Clock Generator is made up of:

- A low-power 32.768 kHz crystal oscillator
- A low-power embedded 64 kHz (typical) RC oscillator generating the 32 kHz source clock
- A 8 to 24 MHz crystal oscillator or a 12 to 48 MHz XRCGB crystal resonator with Bypass mode
- A 12 MHz RC oscillator
- A 480 MHz UTMI PLL providing a clock for the USB High-speed Device Controller
- A 600 to 1200 MHz programmable PLL (input from 8 to 50 MHz), provides the clock to the processor and to the peripherals
- A 700 MHz fractional-N programmable audio PLL, with 22-bit frequency resolution and two independent programmable post dividers to drive the CLK_AUDIO output pin and the internal peripherals (AUDIOPLLCLK)

The Clock Generator provides the following clocks:

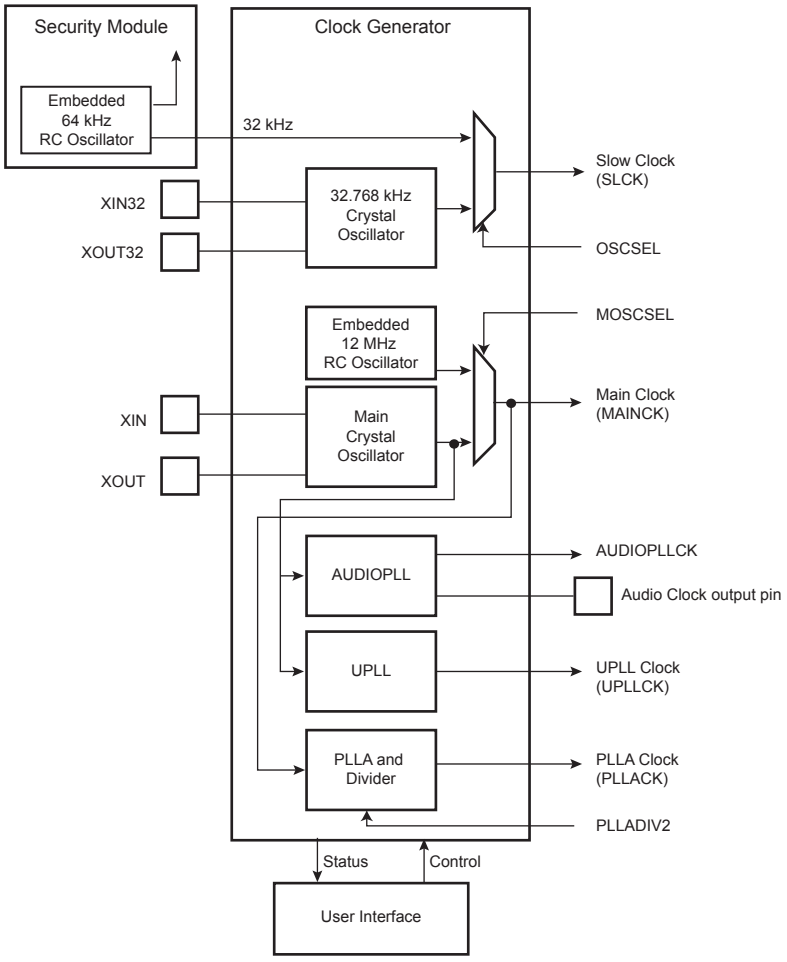
- SLCK—Slow clock. The only permanent clock within the system.
- MAINCK—Output of the Main clock oscillator selection: either 8 to 24 MHz crystal oscillator or 12 MHz RC oscillator
- PLLACK—Output of the divider and the 600 to 1200 MHz programmable PLL (PLLA)
- AUDIOPLLCLK—Output of the first Audio PLL post-divider, with a frequency range from 24 to 125 MHz
- AUDIOPINCLK—Output of the second Audio PLL post-divider, with a frequency range from 8 to 30 MHz
- UPLLCK—Output of the 480 MHz UTMI PLL (UPLL)

The Power Management Controller also provides the following operations on clocks:

- 8 to 24 MHz crystal oscillator clock failure detector
- 32.768 kHz crystal oscillator frequency monitor
- Frequency counter on Main clock and an on-the-fly adjustable 12 MHz RC oscillator frequency

## 31.3 Block Diagram

**Figure 31-1. Clock Generator Block Diagram**




## 31.4 Slow Clock

The Slow Clock Controller embeds a Slow clock generator that is supplied with the VDDBU power supply. As soon as VDDBU is supplied, both the 32.768 kHz crystal oscillator and the embedded 64 kHz (typical) RC oscillator are powered, but only the RC oscillator is enabled.

The Slow clock is generated either by the 32.768 kHz crystal oscillator or by the embedded 64 kHz (typical) RC oscillator divided by two.

The selection of the Slow clock source is made via the OSCSEL bit in the Slow Clock Controller Configuration register (SCKC_CR).

SCKC_CR.OSCSEL and PMC_SR.OSCSELS report which oscillator is selected as the Slow clock source. PMC_SR.OSCSELS informs when the switch sequence initiated by a new value written in SCKC_CR.OSCSEL is done.

### 31.4.1 Embedded 64 kHz (typical) RC Oscillator

By default, the embedded 64 kHz (typical) RC oscillator is enabled and selected as a source of SLCK. The user has to take into account the possible drifts of this oscillator. Refer to the section “DC Characteristics”.

### 31.4.2 32.768 kHz Crystal Oscillator

The Clock Generator integrates a low-power 32.768 kHz crystal oscillator. To use this oscillator, the XIN32 and XOUT32 pins must be connected to a 32.768 kHz crystal. Refer to the section “Electrical Characteristics” for appropriate loading capacitor selection on XIN32 and XOUT32.

Note that the user is not obliged to use the 32.768 kHz crystal oscillator and can use the 64 kHz (typical) RC oscillator instead.

The 32.768 kHz crystal oscillator provides a more accurate frequency than the 64 kHz (typical) RC oscillator.

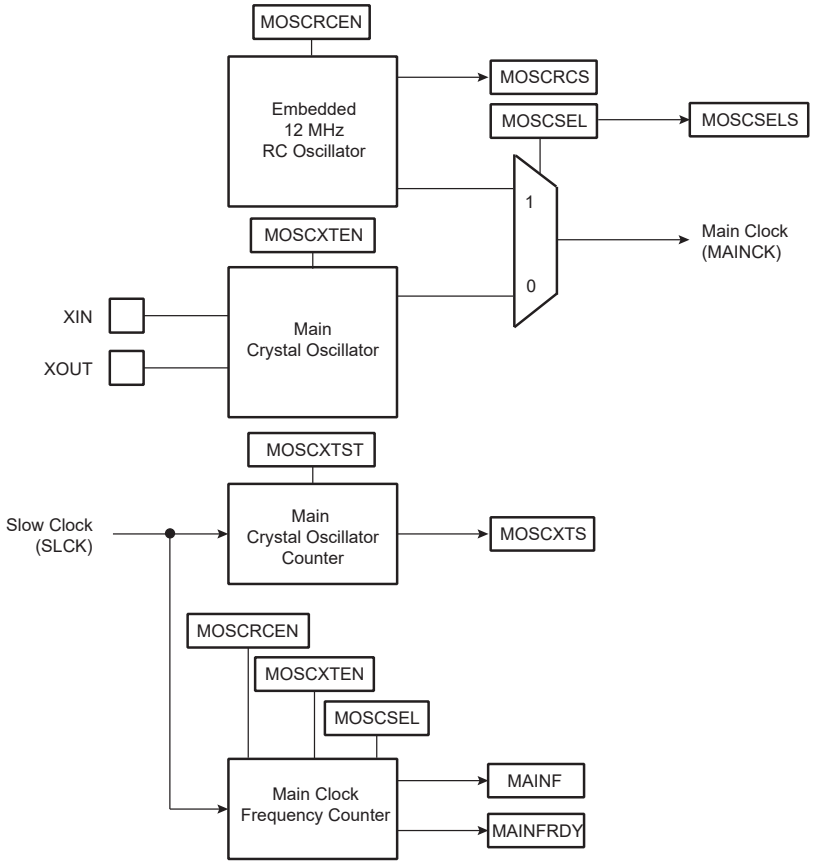
To select the 32.768 kHz crystal oscillator as the source of the Slow clock, the bit SCKC_CR.OSCSEL must be set. This results in a sequence which enables the 32.768 kHz crystal oscillator. The switch of the Slow clock source is glitch-free.

## 31.5 Main Clock

The Main clock has two sources:

- a 12 MHz RC oscillator with a fast start-up time and used at start-up
- a 8 to 24 MHz crystal oscillator with Bypass mode

**Figure 31-2. Main Clock Block Diagram**




### 31.5.1 12 MHz RC Oscillator

After reset, the 12 MHz RC oscillator is enabled and selected as the source of MAINCK and MCK. MCK is the default clock selected to start up the system.

Refer to the table “DC Characteristics”.

The software can disable or enable the 12 MHz RC oscillator with the MOSCRCEN bit in the Clock Generator Main Oscillator register (CKGR_MOR).

When disabling the Main clock by clearing CKGR_MOR.MOSCRCEN, PMC_SR.MOSCRCS is automatically cleared, indicating the Main clock is OFF.

Setting the MOSCRCS bit in the Power Management Controller Interrupt Enable register (PMC_IER) triggers an interrupt to the processor.

### 31.5.2 12 MHz RC Oscillator Clock Frequency Adjustment

It is possible for the user to adjust the 12 MHz RC oscillator frequency through the PMC Oscillator Calibration Register (PMC_OCR). By default, PMC_OCR.SEL is low, so the RC oscillator is driven with fuse calibration bits which are programmed during the chip production.

The user can adjust the trimming of the 12 MHz RC oscillator through PMC_OCR in order to obtain more accurate frequency (to compensate derating factors such as temperature and voltage).

In order to calibrate the main oscillator frequency, SEL must be set to 1 and a correct frequency value must be configured in the CAL field.

It is possible to restart, at anytime, a measurement of the frequency of the selected clock by means of the RCMEAS bit in the Clock Generator Main Clock Frequency register (CKGR_MCFR). Thus, when MAINFRDY flag reads 1, another read access on CKGR_MCFR provides an image of the frequency of the Main clock on MAINF field. The software can calculate the error with an expected frequency and correct PMC_OCR.CAL accordingly. This may be used to compensate frequency drift due to derating factors such as temperature and/or voltage.

### 31.5.3 8 to 24 MHz Crystal Oscillator

After reset, the 8 to 24 MHz crystal oscillator is disabled and is not selected as the source of MAINCK.

As the source of MAINCK, the 8 to 24 MHz crystal oscillator provides an accurate frequency. The software enables or disables this oscillator in order to reduce power consumption via CKGR_MOR.MOSCXTEN.

When disabling this oscillator by clearing the CKGR_MOR.MOSCXTEN bit, the PMC_SR.MOSCXTS bit is automatically cleared, indicating the 8 to 24 MHz crystal oscillator is off.

When enabling this oscillator, the user must initiate the start-up time counter. This start-up time depends on the characteristics of the external device connected to this oscillator. Refer to the section “Electrical Characteristics” for the start-up time.

When CKGR_MOR.MOSCXTEN and CKGR_MOR.MOSCXTST are written to enable this oscillator, the PMC_SR.MOSCXTS bit is cleared and the counter starts counting down on the Slow clock divided by 8 from the MOSCXTST value. When the counter reaches 0, the PMC_SR.MOSCXTS is set, indicating that the 8 to 24 MHz crystal oscillator is stabilized. Setting MOSCXTS in the PMC Interrupt Mask register (PMC_IMR) triggers an interrupt to the processor.

### 31.5.4 Main Clock Source Selection

The source of the Main clock can be selected from the following:

- embedded 12 MHz RC oscillator
- 8 to 24 MHz crystal oscillator
- an XRCGB crystal resonator

The advantage of the Main RC oscillator is its fast start-up time. By default, this oscillator is selected to start the system and it must be selected prior to entering Wait mode.

The advantage of the Main crystal oscillator is its high level of accuracy.

The selection is made by writing CKGR_MOR.MOSCSEL. The switch of the Main clock source is glitch-free, so there is no need to run out of SLCK or PLLACK in order to change the selection. PMC_SR.MOSCSELS indicates when the switch sequence is done.

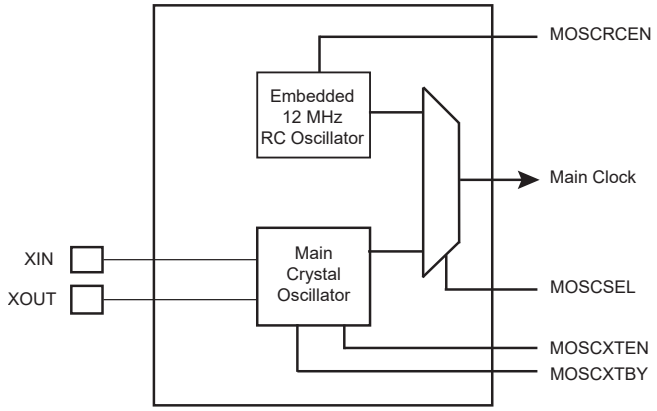
Setting PMC_IMR.MOSCSELS triggers an interrupt to the processor.

The 8 to 24 MHz crystal oscillator can be bypassed by setting CKGR_MOR.MOSCXTBY to accept an external Main clock on XIN (see 31.5.5 Bypassing the 8 to 24 MHz Crystal Oscillator).

MOSCRCEN, MOSCSEL, MOSCXTEN and MOSCXTBY bits are located in the PMC Clock Generator Main Oscillator Register (CKGR_MOR).

After a VDDBU power-on reset, the default configuration is MOSCRCEN = 1, MOSCXTEN = 0 and MOSCSEL = 0, allowing the 12 MHz RC oscillator to start as Main clock.

**Figure 31-3. Main Clock Source Selection**




### 31.5.5 Bypassing the 8 to 24 MHz Crystal Oscillator

Prior to bypassing the 8 to 24 MHz crystal oscillator, the external clock frequency provided on the XIN pin must be stable and within the values specified in the XIN clock characteristics. Refer to the section “Electrical Characteristics”.

The sequence to bypass the crystal oscillator is as follows:

1. Ensure that an external clock is connected on XIN.
2. Enable the bypass by setting CKGR_MOR.MOSCXTBY.
3. Disable the 8 to 24 MHz crystal oscillator by clearing CKGR_MOR.MOSCXTEN.

### 31.5.6 Main Frequency Counter

The main frequency counter measures the Main RC oscillator and the Main crystal oscillator against the SLCK and is managed by CKGR_MCFR.

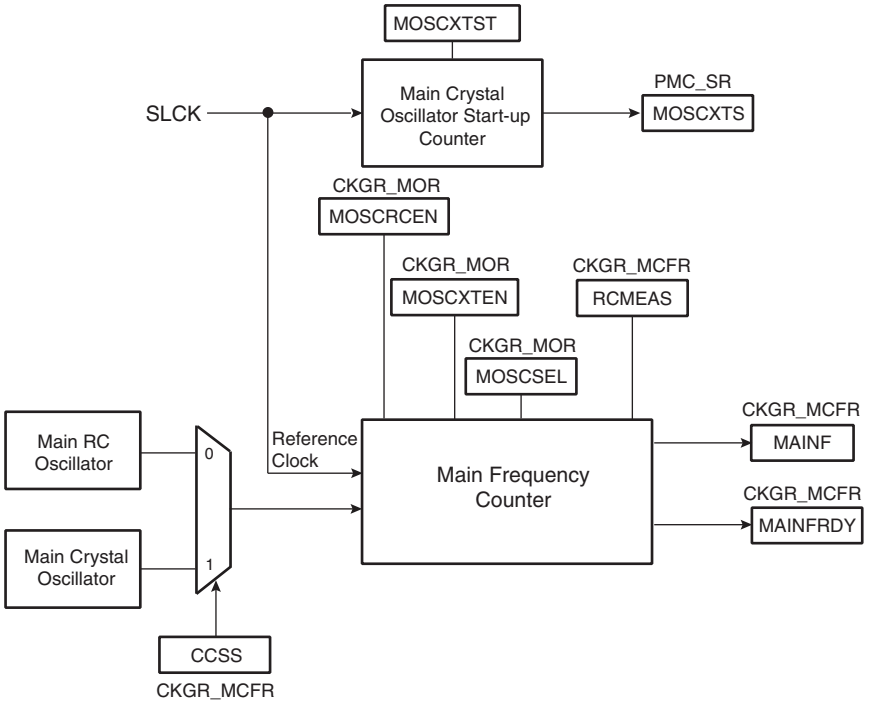
During the measurement period, the main frequency counter increments at the speed of the clock defined by the bit CKGR_MCFR.CCSS.

A measurement is started in the following cases:

- When CKGR_MCFR.RCMEAS is written to ‘1’.
- When the 12 MHz RC oscillator is selected as the source of the Main clock and when this oscillator becomes stable (i.e., when the MOSCRCS bit is set)
- When the 8 to 24 MHz crystal oscillator is selected as the source of the Main clock and when this oscillator becomes stable (i.e., when the MOSCXTS bit is set)
- When the Main clock source selection is modified

The measurement period ends at the 16th falling edge of the Slow clock, CKGR_MCFR.MAINFRDY is set and the counter stops counting. Its value can be read in the CKGR_MCFR.MAINF and gives the number of Main clock cycles during 16 periods of Slow clock, so that the frequency of the 12 MHz RC oscillator or the crystal oscillator can be determined.

**Figure 31-4. Main Frequency Counter Block Diagram**




### 31.5.7 Switching Main Clock Between the RC Oscillator and the Crystal Oscillator

When switching the source of the Main clock between the RC oscillator and the crystal oscillator, both oscillators must be enabled. After completion of the switch, the unused oscillator can be disabled.

If switching to the crystal oscillator, a check must be carried out to ensure that the oscillator is present and that its frequency is valid. Follow the sequence below:

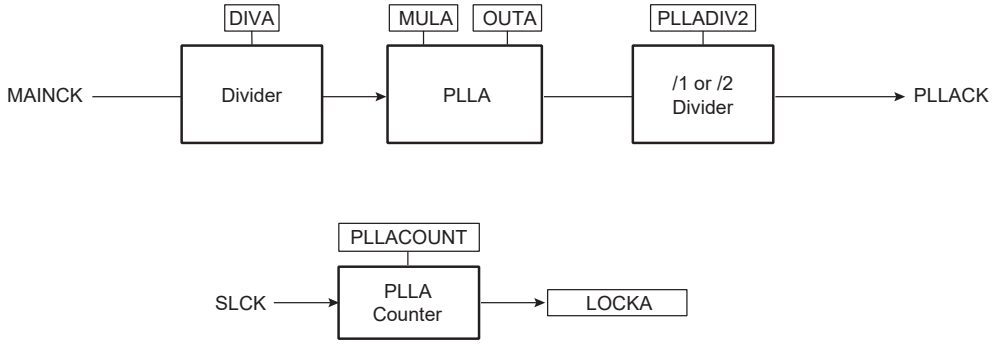
1. Enable the crystal oscillator by setting CKGR_MOR.MOSCXTEN. Configure the CKGR_MOR. MOSCXTST field with the crystal oscillator start-up time as defined in the section “Electrical Characteristics”.
2. Wait for PMC_SR.MOSCXTS flag to rise, indicating the end of a start-up period of the crystal oscillator.
3. Select the crystal oscillator as the source clock of the frequency meter by setting CKGR_MCFR.CCSS
4. Initiate a frequency measurement by setting CKGR_MCFR.RCMEAS.
5. Read CKGR_MCFR.MAINFRDY until its value equals 1.
6. Read CKGR_MCFR.MAINF and compute the value of the crystal frequency.
   – If the MAINF value is valid, the Main clock can be switched to the crystal oscillator.

## 31.6 Divider and PLLA Block

The PLLA embeds an input divider to increase the accuracy of the resulting clock signals. However, the user must respect the PLLA minimum input frequency when programming the divider.

The following figure shows the block diagram of the divider and PLLA block.

**Figure 31-5. Divider and PLLA Block Diagram**




### 31.6.1 Divider and Phase Lock Loop Programming

The PLLA allows multiplication of the divider's outputs. The PLLA clock signal has a frequency that depends on the respective source signal frequency and on the parameters DIVA and MULA. The factor applied to the source signal frequency is (MULA + 1)/DIVA. When MULA is written to 0, the PLLA is disabled and its power consumption is saved. Re-enabling the PLLA can be performed by writing a value higher than 0 in the MUL field.

Whenever the PLLA is re-enabled or one of its parameters is changed, PMC_SR.LOCKA is automatically cleared. The values written in the PLLACOUNT field in the Clock Generator PLLA register (CKGR_PLLAR) are loaded in the PLLA counter. The PLLA counter then decrements at the speed of the Slow clock until it reaches 0. At this time, PMC_SR.LOCKA is set and can trigger an interrupt to the processor. The user has to load the number of Slow clock cycles required to cover the PLLA transient time into CKGR_PLLACOUNT.

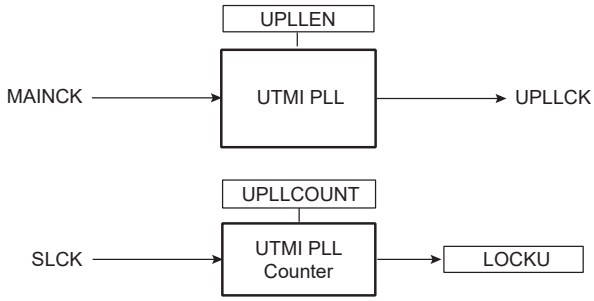
The PLLA clock must be divided by 2 by writing PMC_MCKR.PLLADIV2, if the ratio between Processor clock (PCK) and MCK is 3 (MDIV = 3).

## 31.7 UTMI PLL Clock

The source of the UTMI PLL (UPLL) is the Main clock (MAINCK). MAINCK must select the Main crystal oscillator to meet the frequency accuracy required by USB.

The crystal frequency selection among 12, 16 or 24 MHz must be configured to the correct value in the field SFR_UTMICKTRIM.FREQ, in order to apply the correct multiplier, x40, x30 or x20, respectively.

**Figure 31-6. UTMI PLL Block Diagram**




Whenever the UTMI PLL is enabled by writing UPLLEN in the UTMI Clock register (CKGR_UCKR), PMC_SR.LOCKU is automatically cleared. The values written in CKGR_UCKR.UPLLCOUNT are loaded in the UTMI PLL counter. The UTMI PLL counter then decrements at the speed of the Slow clock divided by 8 until it reaches 0. At this time, the PMC_SR.LOCKU is set in and can trigger an interrupt to the

processor. The user has to load the number of Slow clock cycles required to cover the UTMI PLL transient time into CKGR_UCKR.UPLLCOUNT.

## 31.8 Audio PLL

The Audio PLL is a high-resolution fractional-N digital PLL specifically designed for low jitter operation.

In audio applications, the CLK_AUDIO output pin typically serves as the Master clock frequency generator for external components such as Audio DAC, Audio ADCs, or Audio Codecs, thus saving one crystal on the board.

The reference clock of the Audio PLL is the fast crystal oscillator. The PLL core operating frequency is defined as:

$$f_{\mathrm{AUDIOCORECLK}} = f_{\mathrm{ref}}\left(\mathrm{ND} + 1 + \frac{\mathrm{FRACR}}{2^{22}}\right)$$

where $f_{ref}$ is the frequency of the main crystal oscillator. Refer to the section “PLL Characteristics” for the limits of $f_{AUDIOCORECLK}$.

The PLL core features two post-dividers enabling the generation of two output clock signals, AUDIOPLLCLK and AUDIOPINCLK. AUDIOPLLCLK is dedicated to the PMC and can be sent to the GCLK input of peripherals or to the Programmable clock outputs PCKx. AUDIOPINCLK is dedicated to driving the external audio pin CLK_AUDIO.

The AUDIOPLLCLK frequency is defined by the following formula:

$$f_{\mathrm{AUDIOPLLCLK}} = \frac{f_{\mathrm{AUDIOCORECLK}}}{(\mathrm{QDPMC} + 1)}$$

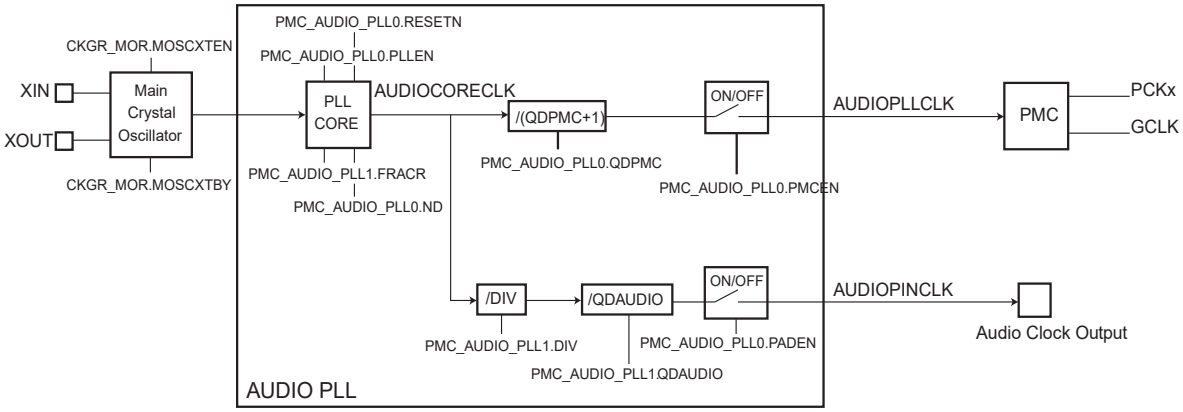
The AUDIOPINCLK frequency is defined by the following formula:

$$f_{\mathrm{AUDIOPINCLK}} = \frac{f_{\mathrm{AUDIOCORECLK}}}{(\mathrm{DIV} \times \mathrm{QDAUDIO})}$$

The typical programming sequence of the audio PLL is the following:

1. Disable the PLL by writing ‘0’ in bits PLLEN and RESETN in the Audio PLL Control register 0 (PMC_AUDIO_PLL0).
2. Release the reset of the PLL by writing ‘1’ in PMC_AUDIO_PLL0.RESETN.
3. Configure the PLL frequency by writing QDPMC and ND in PMC_AUDIO_PLL0, QDAUDIO, DIV and FRACR in PMC_AUDIO_PLL1. ND and FRACR must be configured so as to set AUDIOCORECLK frequency in its authorized range. Refer to the “Electrical Characteristics” section.
4. Enable the PLL by writing ‘1’ in PMC_AUDIO_PLL0.PLLEN, PMC_AUDIO_PLL0.PADEN and PMC_AUDIO_PLL0.PMCEN.
5. Wait for the start-up time of this PLL. Refer to the “Electrical Characteristics” section.
6. If needed, ND or FRACR can be adjusted at any time. The typical frequency settling time of this PLL is indicated in the “Electrical Characteristics” section.

**Figure 31-7. Audio PLL**



# 32. Power Management Controller (PMC)

## 32.1 Description

The Power Management Controller (PMC) optimizes power consumption by controlling all system and user peripheral clocks. The PMC enables/disables the clock inputs to many of the peripherals and the Core.

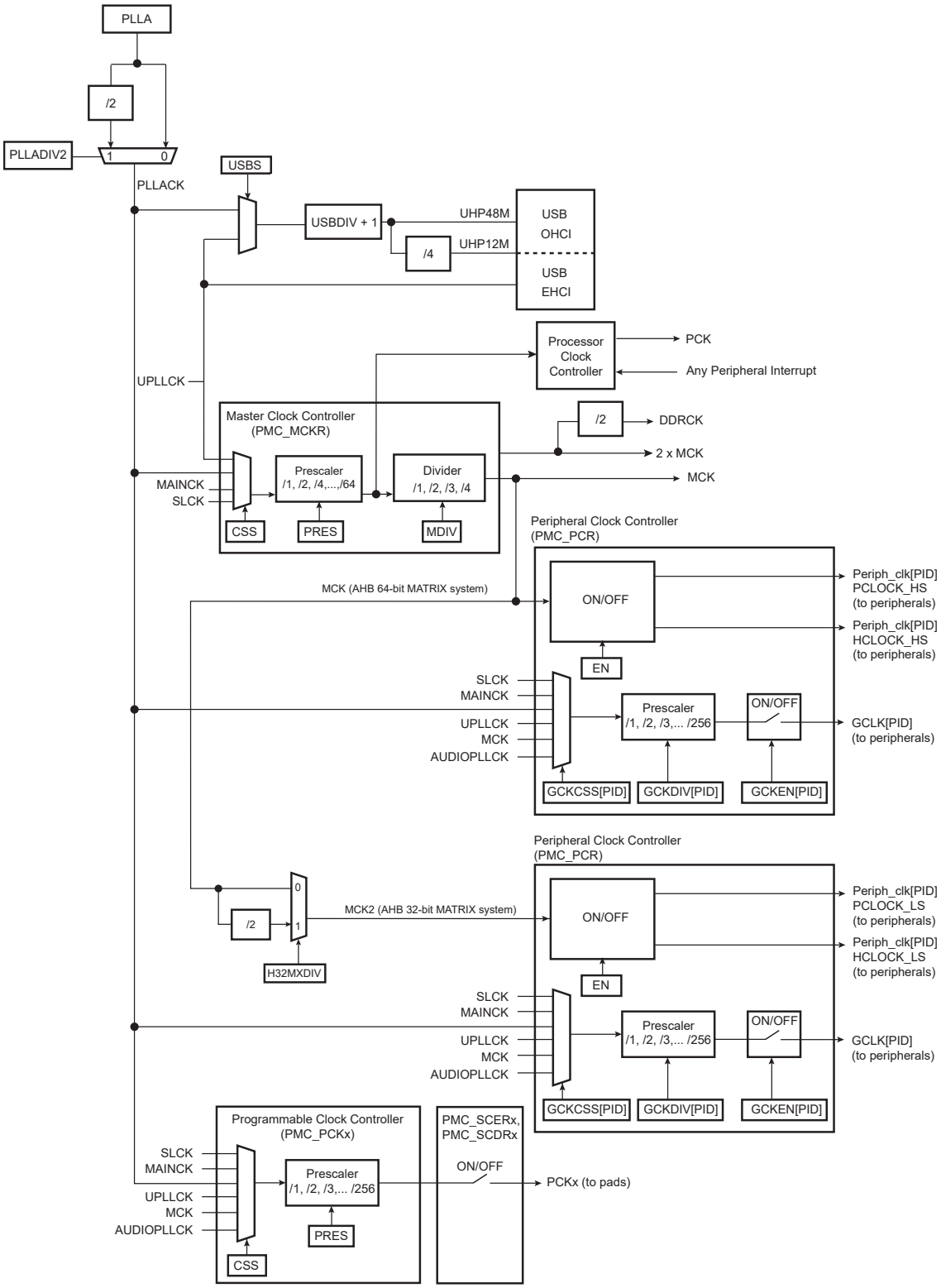
## 32.2 Embedded Characteristics

The Power Management Controller provides the following clocks:

- Master Clock (MCK)—programmable from a few hundred Hz to the maximum operating frequency of the device. It is available to the modules running permanently.
- Processor Clock (PCK)—must be switched off when processor is entering Idle mode
- HS USB Device Clock (UDPCK)
- H64MX Matrix Clock (MCK) and H32MX Matrix Clock (MCK or MCK/2)
- Peripheral Clocks—provided to the embedded peripherals and independently controllable
- Programmable Clock Outputs—can be selected from the clocks provided by the Clock Generator and driven on the PCKx pins
- Generic Clock (GCLK)—for peripherals that can accept a second clock source
- Asynchronous Partial Wake-up (SleepWalking)—for FLEXCOMx, SPIx, TWIx, UARTx and ADC

## 32.3 Block Diagram

**Figure 32-1. General Clock Block Diagram**



## 32.4 Master Clock Controller

The Master Clock Controller provides selection and division of the Master clock (MCK). MCK is the source clock of the peripheral clocks.

The Master clock is selected from one of the clocks provided by the Clock Generator. Selecting the Slow clock provides a Slow clock signal to the whole device. Selecting the Main clock saves power consumption of the PLLs.

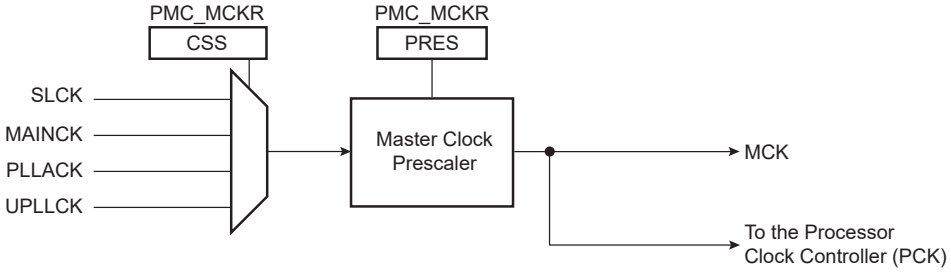
The Master Clock Controller is made up of a clock selector and a prescaler. It also contains a Master clock divider which allows the processor clock to be faster than the Master clock.

The Master clock selection is made by writing the CSS (Clock Source Selection) field in the Master Clock register (PMC_MCKR). The prescaler supports the division by a power of 2 of the selected clock between 1 and 64, and the division by 6. PMC_MCKR.PRES programs the prescaler.

**Note:** It is forbidden to modify MDIV and CSS at the same access. Each field must be modified separately with a wait for MCKRDY flag between the first field modification and the second field modification.

Each time PMC_MCKR is written to define a new Master clock, PMC_SR.MCKRDY is cleared. It reads 0 until the Master clock is established. Then, the MCKRDY bit is set and can trigger an interrupt to the processor. This feature is useful when switching from a high-speed clock to a lower one to inform the software when the change is actually done.

**Figure 32-2. Master Clock Controller**




## 32.5 Processor Clock Controller

The PMC features a Processor Clock (PCK) Controller that implements the processor Idle mode.

The Processor clock can be disabled by executing the WFI (WaitForInterrupt) processor instruction or the WFE (WaitForEvent) processor instruction while the LPM bit is at 0 in the PMC Fast Startup Mode register (PMC_FSMR).

The Processor clock can be disabled by writing the PMC System Clock Disable Register (PMC_SCDR). The status of this clock (at least for debug purposes) can be read in the PMC System Clock Status Register (PMC_SCSR).

The Processor clock is enabled after a reset and is automatically re-enabled by any enabled interrupt. The processor Idle mode is entered by disabling the Processor clock, which is automatically re-enabled by any enabled fast or normal interrupt, or by the reset of the product.

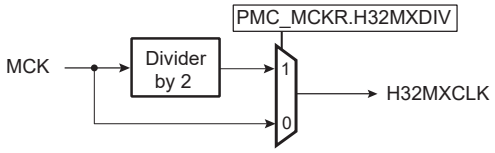
When processor Idle mode is entered, the current instruction is finished before the clock is stopped, but this does not prevent data transfers from other masters of the system bus.

## 32.6 Matrix Clock Controller

The AXI Matrix and H64MX 64-bit Matrix clocks are MCK.

The H32MX 32-bit matrix clock is to be configured as MCK if MCK does not exceed 83 MHz (refer to “Master Clock Characteristics” in the section “Electrical Characteristics”); otherwise, this clock is to be configured as MCK/2. Selection is done with the H32MXDIV bit in PMC Master Clock Register.

**Figure 32-3. H32MX 32-bit Matrix Clock Configuration**




## 32.7 Programmable Clock Controller

The PMC controls three signals to be outputs on external pins PCKx. Each signal can be independently programmed via the Programmable Clock registers (PMC_PCKx).

PCKx can be independently selected between the Slow clock (SLCK), the Master clock (MAINCK), the PLLACK, the UTMI PLL output, the Main clock and the AUDIO PLL (AUDIOPLLCLK) output by writing PMC_PCKx.CSS. Each output signal can also be divided by a factor between 1 and 256 by writing PMC_PCKx.PRES.

Each output signal can be enabled and disabled by writing a ‘1’ in the corresponding bits, PMC_SCER.PCKx and PMC_SCDR.PCKx, respectively. The status of each active programmable output clocks is given in PMC_SCSR.PCKx.

The status flag PMC_SR.PCKRDYx indicates that the Programmable clock programmed in PMC_PCKx is ready.

As the Programmable Clock Controller does not implement glitch prevention when switching clocks, the PCKx must be disabled before any configuration change (clock source and prescaler) and must be re-enabled after the change is performed.

## 32.8 Core and Bus Independent Clocks for Peripherals

The following table lists the peripherals that can operate while the core, bus and peripheral clock frequencies are modified, thus providing communications at a bit rate which is independent for the core/bus/peripheral clock. This mode of operation is possible by using the internally generated independent clock sources.

**Table 32-1. Clock Assignments**

| Peripheral | Specific Clock Requirements |
|---|---|
| GMAC | – |
| UDPHS | – |
| UHPHS | – |
| CLASSD | GCLK |

| .....continued | |
|---|---|
| **Peripheral** | **Specific Clock Requirements** |
| I2SC | GCLK |
| FLEXCOM (USART, SPI, TWI) | GCLK |
| ISC | PCK (pad clock) |
| MCAN | GCLK |
| TC | GCLK |
| ADC | GCLK |

## 32.9 Peripheral and Generic Clock Controller

The PMC controls the clocks of the embedded peripherals by means of the Peripheral Control register (PMC_PCR). With this register, the user can enable and disable the different clocks used by the peripherals:

- Peripheral clocks (periph_clk[..]), routed to every peripheral and derived from the Master clock (MCK), and
- Generic clocks (GCLK[PID]), routed to selected peripherals only (refer to the Peripheral Identifiers table in section Peripherals). These clocks are independent of the core and bus clocks (PCK, MCK and Periph_clk[PID]).

To configure a peripheral's clocks, PMC_PCR.CMD must be written to '1' and PMC_PCR.PID must be written with the index of the corresponding peripheral.

The GCLK[PID] clocks are generated by selection and division of the following sources: SLCK, MAINCK, UPLLCKDIV, PLLACK, AUDIOCKDIV and MCK. The source clock is selected by writing PMC_PCR.GCKCSS and the source clock divider is configured by writing PMC_PCR.GCKDIV. To apply a clock source selection and division factor, PID, CMD and GCKDIV, GCKCSS must be written in a single operation.

As the Generic Clock Controller (GCLK[PID]) does not implement glitch prevention when switching clocks, the GCLK[PID] must be disabled (PMC_PCR.GCKEN=0) before any configuration change (clock source and prescaler) and must be re-enabled after the change is performed.

The peripheral clock (periph_clk[..]) can be enabled or disabled by writing PMC_PCR.EN.

The user can also enable and disable these clocks by configuring the Peripheral Clock Enable (PMC_PCERx) and Peripheral Clock Disable (PMC_PCDRx) registers. The status of the peripheral clock activity can be read in the Peripheral Clock Status registers (PMC_PCSRx).

When a peripheral or a generic clock is disabled, it is immediately stopped. These clocks are disabled after a reset.

To stop a peripheral clock, the application software must wait until the peripheral has executed its last programmed operation before disabling the clock. This is to avoid data corruption or erroneous behavior of the system.

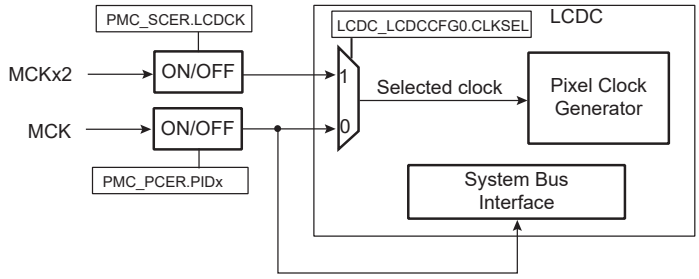
For a peripheral clock (periph_clk[..]), the bit number in PMC_PCERx, PMC_PCDRx, and PMC_PCSRx is the Peripheral Identifier defined at the product level. The bit number corresponds to the interrupt source number assigned to the peripheral.

To read the current clock configuration of a peripheral, PMC_PCR.CMD must be written to ‘0’ and PMC_PCR.PID must be written with the index of the corresponding peripheral regardless of the values of other fields. This write does not modify the configuration of the peripheral. The PMC_PCR can then be read to know the configuration status of the corresponding PID.

## 32.10 LCDC Clock Controller

In order to have more flexibility on the pixel clock, the LCDC can use MCK, or MCKx2 if LCDCK is set in the PMC System Clock Enable Register (PMC_SCER).

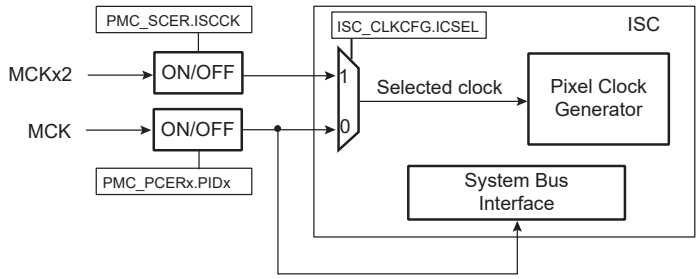
**Figure 32-4. LCDCLK Clock Configuration**




## 32.11 ISC Clock Controller

In order to have more flexibility on the pixel clock, the ISC can use MCK, or MCKx2 if ISCCK is set in the PMC System Clock Enable Register (PMC_SCER).

**Figure 32-5. ISCCLK Clock Configuration**




## 32.12 USB Device and Host Clocks

The USB Device and Host High Speed ports (UDPHS and UHPHS) clocks are enabled by the corresponding PIDx bits in the Peripheral Clock Enable register (PMC_PCERx). To save power on this peripheral when they are not used, the user can set these bits in the Peripheral Clock Disable register (PMC_PCDRx). Corresponding PIDx bits in the Peripheral Clock Status register (PMC_PCSRx) give the status of these clocks.

The PMC also provides the clocks UHP48M and UHP12M to the USB Host OHCI. The USB Host OHCI clocks are controlled by PMC_SCER.UHP. To save power on this peripheral when they are not used, the user can set PMC_SCDR.UHP. PMC_SCSR.UHP gives the status of this clock. The USB host OHCI requires both the 12/48 MHz signal and the Master clock. The USBDIV field in the USB Clock register (PMC_USB) is to be programmed to 9 (division by 10) for normal operations.

To further reduce power consumption the user can stop the UTMI PLL. In this case USB high-speed operations are not possible. Nevertheless, as the USB OHCI Input clock can be selected with PMC_USB.USBS (PLLA or UTMI PLL), OHCI full-speed operation remains possible.

The user must program the USB OHCI Input clock and the USBDIV divider in the PMC_USB register to generate a 48 MHz and a 12 MHz signal with an accuracy of ±0.25%.

The USB clock input is to be defined according to main oscillator via the FREQ field in the UTMI Clock Trimming register (SFR_UTMICKTRIM). Refer to the section “Special Function Registers (SFR)”. This input clock can be 12, 16, or 24 MHz.

## 32.13 DDR2/LPDDR/LPDDR2 Clock Controller

The PMC controls the clocks of the DDR memory.

The DDR clock can be enabled and disabled with the DDRCK bit in PMC_SCER and PMC_SDER, respectively. At reset, the DDR clock is disabled to reduce power consumption.

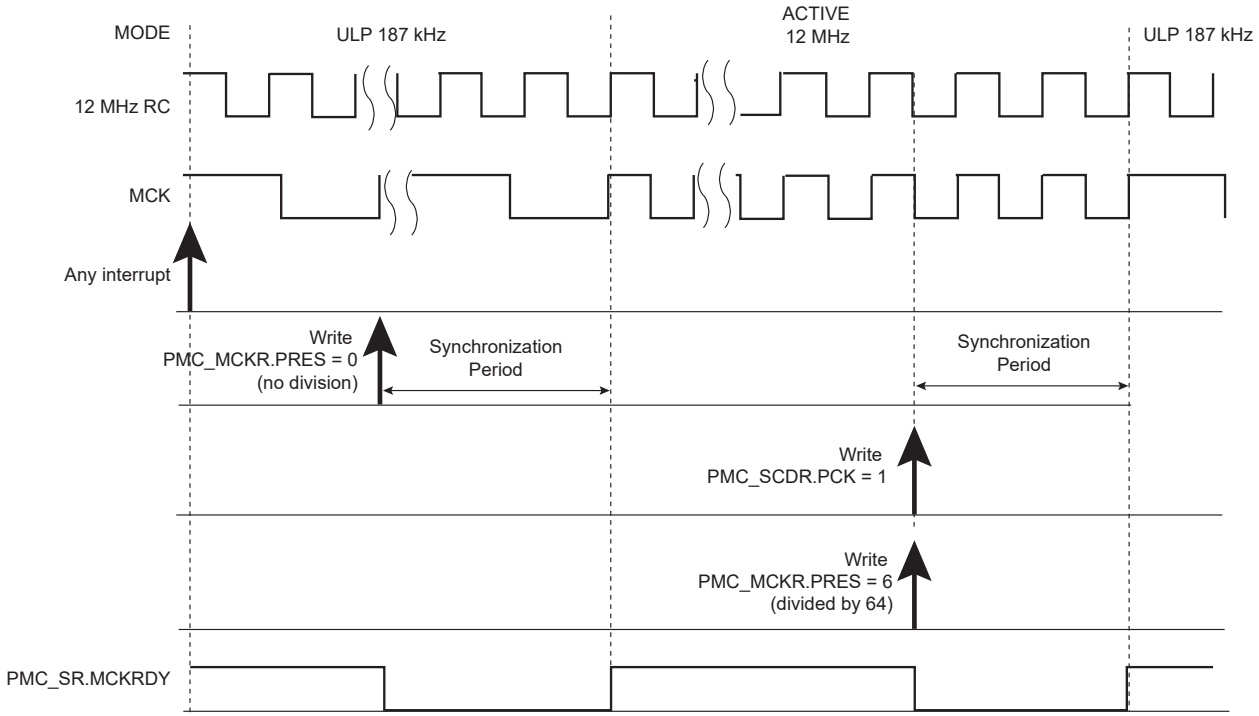
If PMC_MCKR.MDIV = 0 (PCK = MCK), the DDR clock is not available.

To reduce PLLA power consumption, the user can choose UPLLCK as an input clock for the system. In this case, the DDR Controller can drive LPDDR or LPDDR2 at up to 120 MHz.

## 32.14 Fast Start-up from Ultra-Low-Power (ULP) Mode 0

In Ultra-Low-power (ULP) mode 0, the Main clock (MAINCK) must be running, thus either the 12 MHz crystal oscillator or the Fast RC oscillator must be enabled. The lowest power consumption that can be achieved in ULP Mode 0, can be obtained when dividing the selected oscillator frequency by 64 by writing PMC_MCKR.PRES to 6. Any interrupt exits the system from ULP Mode. The software must write PMC_MCKR.PRES to 1 to provide MCK with the fastest clock. If the PLL is used, the start-up procedure must be done prior to writing PMC_MCKR.PRES to 1. The following figure illustrates an example of start-up phase from ULP Mode 0 without use of PLL.

**Figure 32-6. Fast Start-up from Ultra-Low-Power Mode 0**




**WARNING** The duration of the WKUPx pins active level must be greater than four MAINCK cycles.

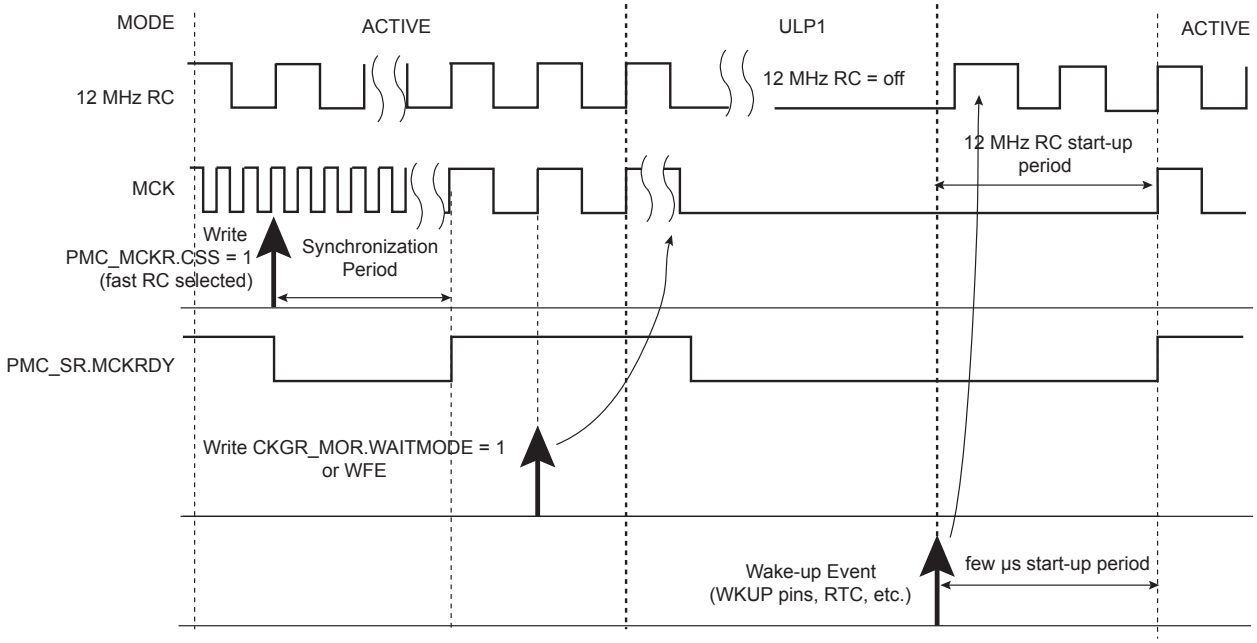
## 32.15 Fast Start-up from Ultra-Low-Power (ULP) Mode 1

The device allows the processor to restart in less than 10 µs while the device exits Ultra-Low-power (ULP) mode 1 only if the C-code function managing the ULP mode 1 entry and exit is linked to and executed from on-chip SRAM.

Prior to instructing the device to enter ULP mode 1, the RC oscillator must be selected as the Master clock source (PMC_MCKR.CSS must be written to 1, wait for PMC_SR.MCKRDY to be set) and the internal sources of wake-up must be cleared. In addition, it must be checked that none of the enabled external wake-up inputs (WKUP) hold an active polarity.

The system enters ULP mode 1 either by setting the CKGR_MOR.WAITMODE, or by executing the WaitForEvent (WFE) instruction of the processor while PMC_FSMR.LPM is at 1. Immediately after setting the WAITMODE bit or using the WFE instruction, wait for PMC_SR.MCKRDY to be set. See the following figure.

**Figure 32-7. Fast Start-up from ULP Mode 1**



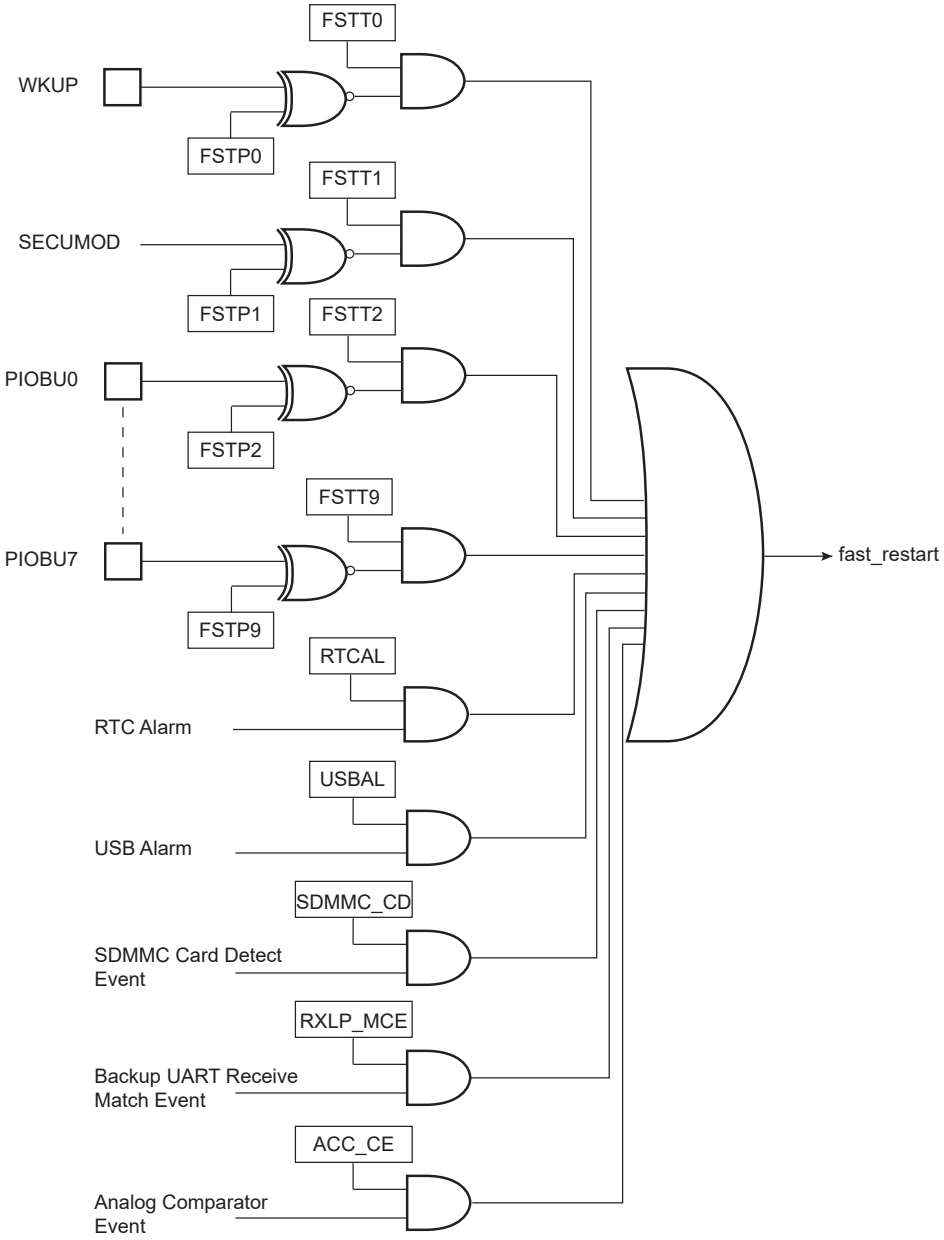

A fast start-up is enabled upon any of the following events:

- detection of a programmed level on one of the nine wake-up inputs (WKUP, PIOBUx)
- an active alarm from the RTC
- a resume from the USB Controller
- SDMMC card detect
- backup UART (RXLP) received character comparison match
- an analog comparison (ACC)
- any SleepWalking event coming from TWI, FLEXCOMx, SPI, ADC

The polarity of the nine wake-up inputs is programmable by writing the PMC Fast Startup Polarity register (PMC_FSPR). All the fast restart event sources except SleepWalking can be individually enabled/disabled by writing in PMC_FSMR. SleepWalking events can be individually enabled/disabled by writing in PMC_SLPWK_ERx/PMC_SLPWK_DRx (see Asynchronous Partial Wake-up (SleepWalking)).

The fast start-up circuitry, as shown in the following figure, is fully asynchronous and provides a fast start-up signal to the PMC. As soon as the fast start-up signal is asserted, the embedded 12 MHz RC oscillator restarts automatically.

**Figure 32-8. Fast Start-up Circuitry**




The PMC user interface does not provide the source of the fast start-up, but the user can recover this information by reading the PIO Controller and the status registers of the RTC, ACC, RXLP, and USB Controller.

## 32.16 Asynchronous Partial Wake-up (SleepWalking)

### 32.16.1 Description

The asynchronous partial wake-up (SleepWalking) wakes up a peripheral in a fully asynchronous way when activity is detected on the communication line. Moreover, under some user configurable conditions, the asynchronous partial wake-up can trigger an exit of the system from ULP mode 1 (full system wake-up).

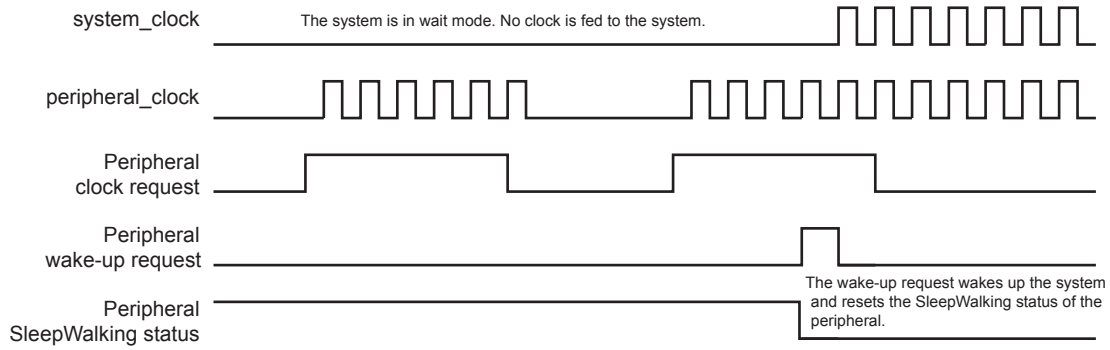
The asynchronous partial wake-up function automatically manages the peripheral clock. It improves the overall power consumption of the system by clocking peripherals only when needed.

Only the following peripherals can be configured with asynchronous partial wake-up: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

The peripheral selected for asynchronous partial wake-up must be first configured so that its clock is enabled by setting the appropriate PIDx bit in PMC_PCERx.

When the system is in ULP mode 1, all clocks of the system (except SLCK) are stopped. When an asynchronous clock request from a peripheral occurs, the PMC partially wakes up the system to feed the clock only to this peripheral. The rest of the system is not fed with the clock, thus optimizing power consumption. Finally, depending on user-configurable conditions, the peripheral either wakes up the whole system if these conditions are met or stops the peripheral clock until the next clock request. If a wake-up request occurs, the Asynchronous Partial Wake-up mode is automatically disabled until the user instructs the PMC to enable asynchronous partial wake-up. This is done by setting PIDx in the PMC SleepWalking Enable register (PMC_SLPWK_ER).

**Figure 32-9. SleepWalking During Ultra-Low-Power Mode 1**

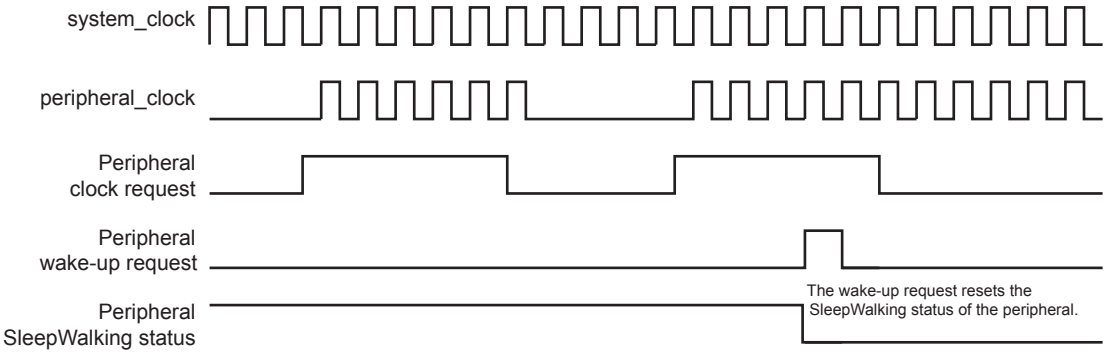



When the system is in Active mode, peripherals enabled for asynchronous partial wake-up have their respective clocks stopped until the peripherals request a clock. When a peripheral requests the clock, the PMC provides the clock without CPU intervention.

The triggering of the peripheral clock request depends on conditions which can be configured for each peripheral. If these conditions are met, the peripheral asserts a request to the PMC. The PMC disables the Asynchronous Partial Wake-up mode of the peripheral and provides the clock to the peripheral until the user instructs the PMC to re-enable partial wake-up on the peripheral. This is done by setting PIDx in PMC_SLPWK_ER.

If the conditions are not met, the peripheral clears the clock request and PMC stops the peripheral clock until the clock request is re-asserted by the peripheral.

**Figure 32-10. SleepWalking During Active Mode**




### 32.16.2 Configuration Procedure

Before configuring the asynchronous partial wake-up (SleepWalking) function of a peripheral, check that the peripheral clock is enabled (the PIDx bit in the PMC Peripheral Clock Status register (PMC_PCSR) must be set).

The procedure to enable the asynchronous partial wake-up (SleepWalking) function of a peripheral is the following:

1. Check that the corresponding PIDx bit in the PMC SleepWalking Activity Status register (PMC_SLPWK_ASR) is cleared. This ensures that the peripheral has no activity in progress.
2. Enable the asynchronous partial wake-up function of the peripheral by writing a one to the corresponding PIDx bit in PMC_SLPWK_ER.
3. Check that the corresponding PIDx bit in PMC_SLPWK_ASR is cleared. This ensures that no activity has started during the enable phase.
4. In PMC_SLPWK_ASR, if the corresponding PIDx bit is set, the asynchronous partial wake-up function must be immediately disabled by writing a one to the PIDx bit in the PMC SleepWalking Disable register (PMC_SLPWK_DR). Wait for the end of peripheral activity before reinitializing the procedure.
   If the corresponding PIDx bit is cleared, then the peripheral clock is disabled and the system can now be placed in ULP mode 1.

Before entering ULP mode 1, check that the AIP bit in the PMC SleepWalking Activity In Progress register (PMC_SLPWK_AIPR) is cleared. This ensures that none of the peripherals has any activity in progress.

**Note:** When asynchronous partial wake-up (SleepWalking) of a peripheral is enabled and the core is running (system not in ULP mode 1), the peripheral must not be accessed before a wake-up of the peripheral is performed.

## 32.17 Main Crystal Oscillator Failure Detection

The Main crystal oscillator failure detector monitors the 8 to 24 MHz crystal oscillator or ceramic resonator-based oscillator to identify a possible failure of this oscillator.

The clock failure detector can be enabled or disabled by configuring CKGR_MOR.CFDEN. The detector is also disabled in either of the following cases:

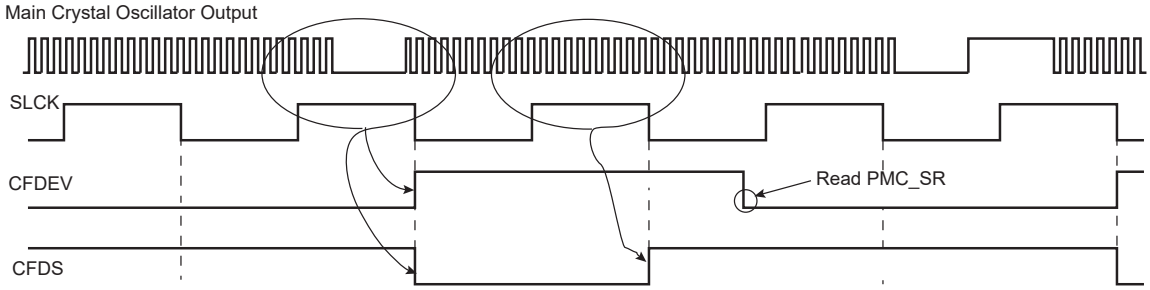
- after a VDDCORE reset
- when the oscillator is disabled (CKGR_MOR.MOSCXTEN = 0)

A failure is detected by means of a counter incrementing on the main oscillator clock edge and detection logic is triggered by the 32 kHz generated by the 64 kHz (typical) RC oscillator. This oscillator is automatically enabled when CKGR_MOR.CFDEN = 1.

The counter is cleared when the 32 kHz generated by the 64 kHz (typical) RC oscillator clock signal is low, and enabled when the signal is high. Thus, the failure detection time is one RC oscillator period. If, during the high level period of the 32 kHz generated by the 64 kHz (typical) RC oscillator clock signal, less than eight 8 to 24 MHz crystal oscillator clock periods have been counted, then a failure is reported.

If a failure of the Main clock is detected, bit PMC_SR.CFDEV indicates a failure event and generates an interrupt if the corresponding interrupt source is enabled. The interrupt remains active until a read occurs in PMC_SR. The user can know the status of the clock failure detection at any time by reading bit PMC_SR.CFDS.

**Figure 32-11. Clock Failure Detection (Example)**




Note: Ratio of clock periods is for illustration purposes only.

If the 8 to 24 MHz crystal oscillator or ceramic resonator-based oscillator is selected as the source clock of MAINCK (CKGR_MOR.MOSCSEL = 1), and if MCK source is PLLACK or UPLLCK (PMC_MCKR.CSS = 2 or 3), a clock failure detection automatically forces MAINCK to be the source clock for the Master clock (MCK). Then, regardless of the PMC configuration, a clock failure detection automatically forces the 12 MHz RC oscillator to be the source clock for MAINCK. If this oscillator is disabled when a clock failure detection occurs, it is automatically re-enabled by the clock failure detection mechanism.

It takes two 32 kHz (typical) clock cycles to detect and switch from the 8 to 24 MHz crystal oscillator to the 12 MHz RC oscillator if the source Master clock (MCK) is Main clock (MAINCK), or three 32 kHz (typical) cycles if the source of MCK is PLLACK or UPLLCK.

A clock failure detection activates a fault output that is connected to the Pulse Width Modulator (PWM) Controller. With this connection, the PWM controller is able to force its outputs and to protect the driven device, if a clock failure is detected.

The user can know the status of the clock failure detector at any time by reading bit PMC_SR.FOS.

This fault output remains active until the defect is detected and until it is cleared by the bit FOCLR in the PMC Fault Output Clear register (PMC_FOCR).

## 32.18 32.768 kHz Crystal Oscillator Frequency Monitor

The frequency of the 32.768 kHz crystal oscillator can be monitored by means of logic driven by the 12 MHz RC oscillator known as a reliable clock source. This function is enabled by configuring the XT32KFME bit of CKGR_MOR.

The error flag XT32KERR in PMC_SR is asserted when the 32.768 kHz crystal oscillator frequency is out of the ±10% nominal frequency value (i.e., 32.768 kHz). The error flag can be cleared only if the Slow clock frequency monitoring is disabled.

The monitored clock frequency is declared invalid if at least four consecutive clock period measurement results are over the nominal period ±10%.

Due to the possible frequency variation of the embedded 12 MHz RC oscillator acting as reference clock for the monitor logic, any Slow clock crystal frequency deviation over ±10% of the nominal frequency is systematically reported as an error by means of PMC_SR.XT32KERR. Between -1% and -10% and +1% and +10%, the error is not systematically reported.

Thus, only a crystal running at a 32.768 kHz frequency ensures that the error flag is not asserted. The permitted drift of the crystal is 10000 ppm (1%), which allows any standard crystal to be used.

The error flag can be defined as an interrupt source of the PMC by setting PMC_IER.XT32KERR.

## 32.19 Programming Sequence

1. If the 8 to 24 MHz crystal oscillator is not required, PLL can be directly configured (begin with Step 6. or Step 7.) else this oscillator must be started (begin with Step 2.).
2. Enable the 8 to 24 MHz crystal oscillator by setting CKGR_MOR.MOSCXTEN. The user can define a start-up time. This can be achieved by writing a value in CKGR_MOR.MOSCXTST. Once this register has been correctly configured, the user must wait for PMC_SR.MOSCXTS to be set. This can be done either by polling MOSCXTS or by waiting for the interrupt line to be raised if the associated interrupt source (MOSCXTS) has been enabled in PMC_IER.
3. Switch the MAINCK to the 8 to 24 MHz crystal oscillator by setting CKGR_MOR.MOSCSEL.
4. Wait for PMC_SR.MOSCSELS to be set to ensure the switchover is complete.
5. Check the Main clock frequency:
   The Main clock frequency can be measured via CKGR_MCFR.

   Read CKGR_MCFR until the MAINFRDY field is set, after which the user can read the field CKGR_MCFR.MAINF by performing an additional read. This provides the number of Main clock cycles that have been counted during a period of 16 Slow clock cycles.

   If MAINF = 0, switch the MAINCK to the 12 MHz RC oscillator by clearing CKGR_MOR.MOSCSEL.
   If MAINF ≠ 0, proceed to Step 6.
6. Set the PLLA and divider (if not required, proceed to Step 7.)
   All parameters needed to configure PLLA and the divider are located in CKGR_PLLAR.

   The MULA field is the PLLA multiplier factor. This parameter can be programmed between 0 and 127. If MULA is cleared, PLLA is turned off, otherwise the PLLA output frequency is PLLA input frequency multiplied by (MULA + 1).

   The PLLACOUNT field specifies the number of Slow clock cycles before LOCKA bit is set in PMC_SR after CKGR_PLLAR has been written.

   Once CKGR_PLLAR has been written, the user must wait for the LOCKA bit to be set in PMC_SR. This can be done either by polling LOCKA in PMC_SR or by waiting for the interrupt line to be raised if the associated interrupt source (LOCKA) has been enabled in PMC_IER. All parameters in CKGR_PLLAR can be programmed in a single write operation. If at some stage parameter MULA or DIVA is modified, LOCKA bit goes low to indicate that PLLA is not yet ready. When PLLA is locked, LOCKA is set again.

   The user must wait for the LOCKA bit to be set before using the PLLA output clock.
7. Set High-speed PLL (UPLL) for UTMI

The UTMI PLL is enabled by setting CKGR_UCKR.UPLLEN. In some cases, it may be preferable to define a start-up time. This can be achieved by writing a value in CKGR_UCKR.PLLCOUNT.

Once this register has been correctly configured, the user must wait for PMC_SR.LOCKU to be set. This can be done either by polling LOCKU in PMC_SR or by waiting for the interrupt line to be raised if the associated interrupt source (LOCKU) has been enabled in PMC_IER.

8. Select Master Clock and Processor Clock
The Master clock and the Processor clock are configurable via PMC_MCKR.

The CSS field is used to select the clock source of the Master clock and Processor clock dividers. By default, the selected clock source is the Main clock.

The PRES field is used to define the Processor clock and Master clock prescaler. The user can choose between different values from 1 to 256). Prescaler output is the selected clock source frequency divided by the PRES value.

The MDIV field is used to define the Master clock divider. It is possible to choose between different values (0, 1, 2, 3). The Master clock output is Processor clock frequency divided by 1, 2, 3 or 4, depending on the value programmed in MDIV.

The PMC PLLA clock input must be divided by 2 by writing the PLLADIV2 bit if MDIV is set to 3.

By default, MDIV and PLLLADIV2 are cleared, which indicates that Processor clock is equal to the Master clock.

Once PMC_MCKR has been written, the user must wait for the MCKRDY bit to be set in PMC_SR. This can be done either by polling MCKRDY in PMC_SR or by waiting for the interrupt line to be raised if the associated interrupt source (MCKRDY) has been enabled in PMC_IER.

PMC_MCKR must not be programmed in a single write operation. The programming sequence for PMC_MCKR is the following:

If a new value for CSS field corresponds to PLL clock,

a. Program PMC_MCKR.PRES.

b. Wait for PMC_SR.MCKRDY to be set.

c. Program PMC_MCKR.MDIV.

d. Wait for PMC_SR.MCKRDY to be set.

e. Program PMC_MCKR.CSS.

f. Wait for PMC_SR.MCKRDY to be set.

If a new value for CSS field corresponds to Main clock or Slow clock,

a. Program PMC_MCKR.CSS.

b. Wait for PMC_SR.MCKRDY to be set.

c. Program PMC_MCKR.PRES.

d. Wait for PMC_SR.MCKRDY to be set.

If CSS, MDIV or PRES are modified at some stage, the MCKRDY bit goes low to indicate that the Master clock and the Processor clock are not yet ready. The user must wait for the MCKRDY bit to be set again before using the Master and Processor clocks.

**Note:** If PLLA clock was selected as the Master clock and the user decides to modify it by writing in CKGR_PLLR, the MCKRDY flag goes low while PLL is unlocked. Once PLL is locked again, LOCKA goes high and MCKRDY is set.
While PLL is unlocked, the Master clock selection is automatically changed to Slow clock. For further information, see Clock Switching Waveforms.

Code Example:

write_register(PMC_MCKR,0x00000001)

wait (MCKRDY=1)

write_register(PMC_MCKR,0x00000011)

wait (MCKRDY=1)

The Master clock is Main clock divided by 2.

The Processor clock is the Master clock.

9. Select Programmable Clocks
Programmable clocks can be enabled and/or disabled via PMC_SCER and PMC_SCDR. 3 programmable clocks can be used. PMC_SCSR indicates which programmable clock is enabled. By default all programmable clocks are disabled.

   PMC_PCKx registers are used to configure programmable clocks.

   The PMC_PCKx.CSSfield selects the programmable clock divider source. Five clock options are available: Main clock, Slow clock, Master clock, PLLACK, UPLLCK. The Slow clock is the default clock source.

   The PRES field is used to control the programmable clock prescaler. It is possible to choose among different values (from 1 to 256). Programmable clock output is prescaler input divided by PRES parameter. By default, the PRES value is cleared which means that PCKx is equal to Slow clock.

   Once the PMC_PCKx register has been configured, The corresponding programmable clock must be enabled and the user is constrained to wait for the PCKRDYx bit to be set in PMC_SR. This can be done either by polling PCKRDYx in PMC_SR or by waiting for the interrupt line to be raised if the associated interrupt source (PCKRDYx) has been enabled in PMC_IER. All parameters in PMC_PCKx can be programmed in a single write operation.

   If the CSS and PRES parameters are to be modified, the corresponding programmable clock must be disabled first. The parameters can then be modified. Once this has been done, the user must re-enable the programmable clock and wait for the PCKRDYx bit to be set.

10. Enable Peripheral Clocks
Once all of the previous steps have been completed, the peripheral clocks can be enabled and/or disabled via PMC_PCERx and PMC_PCDRx.

## 32.20 Clock Switching Details

### 32.20.1 Master Clock Switching Timings

The following tables give the worst case timings required for the Master clock to switch from one selected clock to another one. This is in the event that the prescaler is deactivated. When the prescaler is activated, an additional time of 64 clock cycles of the new selected clock has to be added.

**Table 32-2. Clock Switching Timings (Worst Case)**

| To | From | | |
|---|---|---|---|
| | Main Clock | SLCK | PLL Clock |
| Main Clock | – | 4 × SLCK +<br>2.5 × Main Clock | 3 × PLL Clock +<br>4 × SLCK +<br>1 × Main Clock |
| SLCK | 0.5 × Main Clock +<br>4.5 × SLCK | – | 3 × PLL Clock +<br>5 × SLCK |
| PLL Clock | 0.5 × Main Clock +<br>4 × SLCK +<br>PLLCOUNT × SLCK +<br>2.5 × PLL Clock | 2.5 × PLL Clock +<br>5 × SLCK +<br>PLLCOUNT × SLCK | 2.5 × PLL Clock +<br>4 × SLCK +<br>PLLCOUNT × SLCK |

**Note:** PLL designates either the PLLA or the UPLL Clock. PLLCOUNT designates either PLLACOUNT or UPLLCOUNT.

**Table 32-3. Clock Switching Timings Between Two PLLs (Worst Case)**

| To | From | |
|---|---|---|
| | PLLA Clock | UPLL Clock |
| PLLA Clock | 2.5 × PLLA Clock +<br>4 × SLCK +<br>PLLACOUNT × SLCK | 3 × PLLA Clock +<br>4 × SLCK +<br>1.5 × PLLA Clock |
| UPLL Clock | 3 × UPLL Clock +<br>4 × SLCK +<br>1.5 × UPLL Clock | 2.5 × UPLL Clock +<br>4 × SLCK +<br>UPLLCOUNT × SLCK |

### 32.20.2 Clock Switching Waveforms

**Figure 32-12. Switch Master Clock from Slow Clock to PLL Clock**

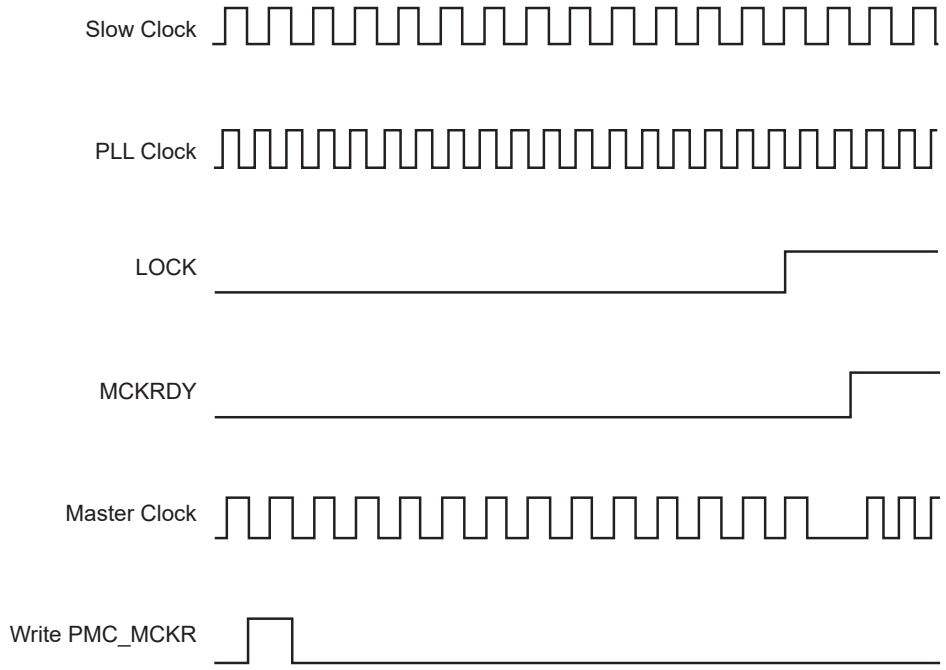



**Figure 32-13. Switch Master Clock from Main Clock to Slow Clock**

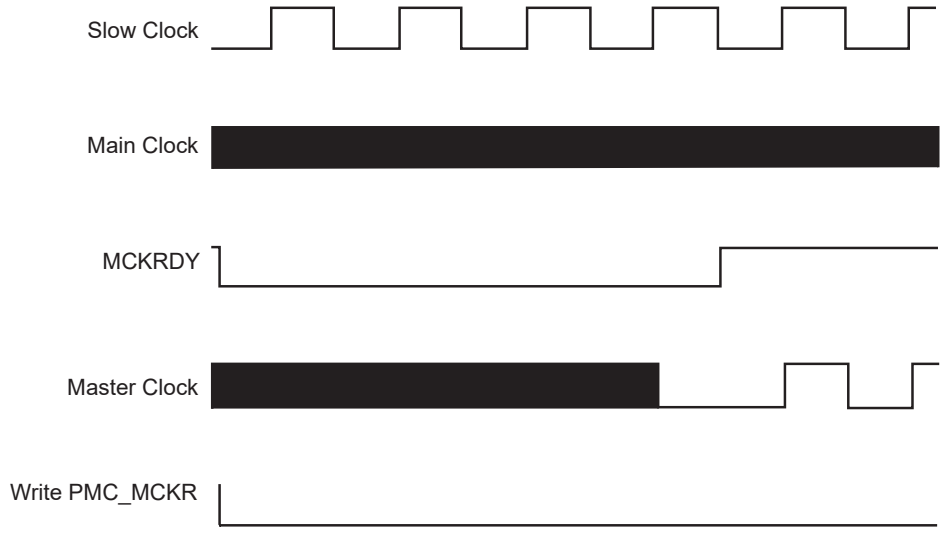


**Figure 32-14. Change PLLA Programming**

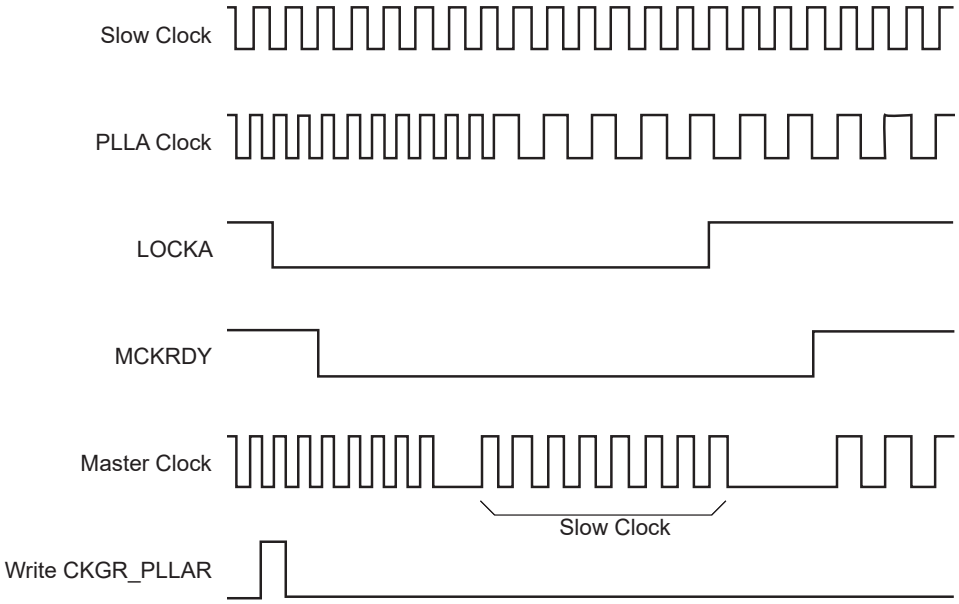
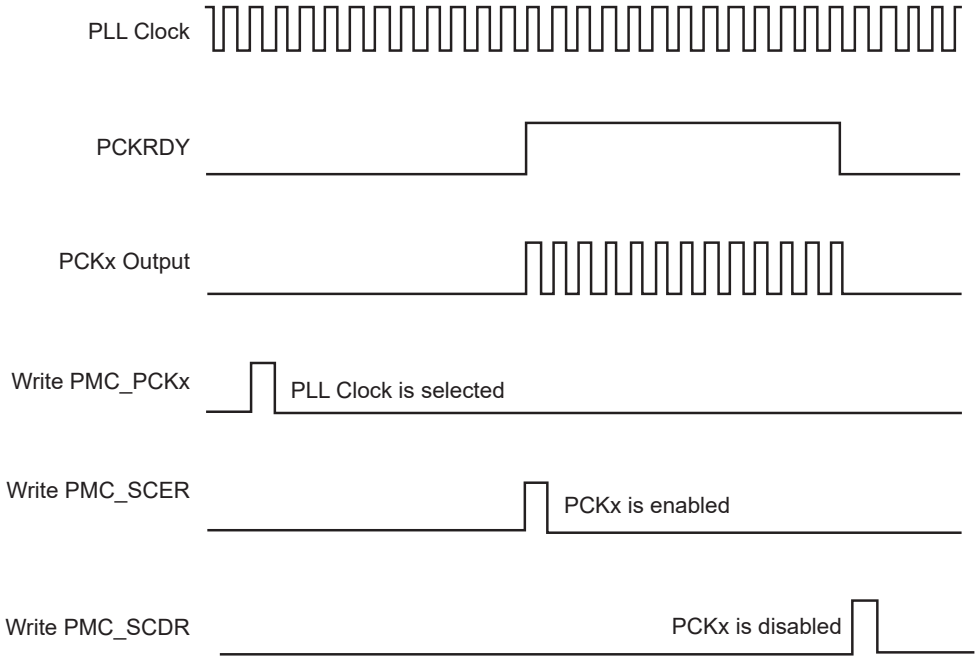



**Figure 32-15. Programmable Clock Output Programming**




## 32.21 Register Write Protection

To prevent any single software error from corrupting PMC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the PMC Write Protection Mode Register (PMC_WPMR).

If a write access to a write-protected register is detected, the WPVS bit in the PMC Write Protection Status Register (PMC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading PMC_WPSR.

The following registers can be write-protected:

- PMC System Clock Enable Register
- PMC System Clock Disable Register
- PMC Peripheral Clock Enable Register 0
- PMC Peripheral Clock Disable Register 0
- PMC Clock Generator Main Oscillator Register
- PMC Clock Generator Main Clock Frequency Register
- PMC Clock Generator PLLA Register
- PMC Master Clock Register
- PMC USB Clock Register
- PMC Fast Startup Polarity Register
- PMC Fast Startup Mode Register
- PLL Charge Pump Current Register
- PMC Oscillator Calibration Register
- PMC SleepWalking Enable Register 0
- PMC SleepWalking Disable Register 1
- PMC SleepWalking Enable Register 1
- PMC SleepWalking Disable Register 1
- PMC SleepWalking Control Register

## 32.22 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | PMC_SCER | 7:0 | UDP | UHP | | | LCDCK | DDRCK | | |
| | | 15:8 | | | | | | PCK2 | PCK1 | PCK0 |
| | | 23:16 | | | | | | ISCCK | | |
| | | 31:24 | | | | | | | | |
| 0x04 | PMC_SCDR | 7:0 | UDP | UHP | | | LCDCK | DDRCK | | PCK |
| | | 15:8 | | | | | | PCK2 | PCK1 | PCK0 |
| | | 23:16 | | | | | | ISCCK | | |
| | | 31:24 | | | | | | | | |
| 0x08 | PMC_SCSR | 7:0 | UDP | UHP | | | LCDCK | DDRCK | | PCK |
| | | 15:8 | | | | | | PCK2 | PCK1 | PCK0 |
| | | 23:16 | | | | | | ISCCK | | |
| | | 31:24 | | | | | | | | |
| 0x0C ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | PMC_PCER0 | 7:0 | PID7 | PID6 | PID5 | PID4 | PID3 | PID2 | | |
| | | 15:8 | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| | | 23:16 | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| | | 31:24 | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| 0x14 | PMC_PCDR0 | 7:0 | PID7 | PID6 | PID5 | PID4 | PID3 | PID2 | | |
| | | 15:8 | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| | | 23:16 | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| | | 31:24 | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| 0x18 | PMC_PCSR0 | 7:0 | PID7 | PID6 | PID5 | PID4 | PID3 | PID2 | | |
| | | 15:8 | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| | | 23:16 | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| | | 31:24 | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| 0x1C | CKGR_UCKR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | UPLLCOUNT[3:0] | | | | | | | UPLLEN |
| | | 31:24 | BIASCOUNT[3:0] | | | | | | | |
| 0x20 | CKGR_MOR | 7:0 | | | ONE | | MOSCRCEN | WAITMODE | MOSCXTBY | MOSCXTEN |
| | | 15:8 | MOSCXTST[7:0] | | | | | | | |
| | | 23:16 | KEY[7:0] | | | | | | | |
| | | 31:24 | | | | | | | CFDEN | MOSCSEL |
| 0x24 | CKGR_MCFR | 7:0 | MAINF[7:0] | | | | | | | |
| | | 15:8 | MAINF[15:8] | | | | | | | |
| | | 23:16 | | | | RCMEAS | | | | MAINFRDY |
| | | 31:24 | | | | | | | | CCSS |
| 0x28 | CKGR_PLLAR | 7:0 | | | | | | | | DIVA |
| | | 15:8 | OUTA[1:0] | | PLLACOUNT[5:0] | | | | | |
| | | 23:16 | MULA[5:0] | | | | | | OUTA[3:2] | |
| | | 31:24 | | | ONE | | | | | MULA[6:6] |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x2C ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | PMC_MCKR | 7:0 | | PRES[2:0] | | | | | CSS[1:0] | |
| | | 15:8 | | | | PLLADIV2 | | | MDIV[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | H32MXDIV |
| 0x34 ... 0x37 | Reserved | | | | | | | | | |
| 0x38 | PMC_USB | 7:0 | | | | | | | | USBS |
| | | 15:8 | | | | | USBDIV[3:0] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x3C ... 0x3F | Reserved | | | | | | | | | |
| 0x40 | PMC_PCK0 | 7:0 | PRES[3:0] | | | | | CSS[2:0] | | |
| | | 15:8 | | | | | PRES[7:4] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x44 | PMC_PCK1 | 7:0 | PRES[3:0] | | | | | CSS[2:0] | | |
| | | 15:8 | | | | | PRES[7:4] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x48 | PMC_PCK2 | 7:0 | PRES[3:0] | | | | | CSS[2:0] | | |
| | | 15:8 | | | | | PRES[7:4] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x4C ... 0x5F | Reserved | | | | | | | | | |
| 0x60 | PMC_IER | 7:0 | | LOCKU | | | MCKRDY | | LOCKA | MOSCXTS |
| | | 15:8 | | | | | | PCKRDYx[2:0] | | |
| | | 23:16 | | | | | | CFDEV | MOSCRCS | MOSCSELS |
| | | 31:24 | | | | | | | | |
| 0x64 | PMC_IDR | 7:0 | | LOCKU | | | MCKRDY | | LOCKA | MOSCXTS |
| | | 15:8 | | | | | | PCKRDYx[2:0] | | |
| | | 23:16 | | | | | | CFDEV | MOSCRCS | MOSCSELS |
| | | 31:24 | | | | | | | | |
| 0x68 | PMC_SR | 7:0 | OSCSELS | LOCKU | | | MCKRDY | | LOCKA | MOSCXTS |
| | | 15:8 | | | | | | PCKRDYx[2:0] | | |
| | | 23:16 | | | | FOS | CFDS | CFDEV | MOSCRCS | MOSCSELS |
| | | 31:24 | | | | | | | | GCKRDY |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x6C | PMC_IMR | 7:0 | | | | | MCKRDY | | LOCKA | MOSCXTS |
| | | 15:8 | | | | | | PCKRDYx[2:0] | | |
| | | 23:16 | | | | | | CFDEV | MOSCRCS | MOSCSELS |
| | | 31:24 | | | | | | | | |
| 0x70 | PMC_FSMR | 7:0 | FSTT7 | FSTT6 | FSTT5 | FSTT4 | FSTT3 | FSTT2 | FSTT1 | FSTT0 |
| | | 15:8 | | | | | | FSTT10 | FSTT9 | FSTT8 |
| | | 23:16 | | | | LPM | SDMMC_CD | USBAL | RTCAL | |
| | | 31:24 | | | | | | | ACC_CE | RXLP_MCE |
| 0x74 | PMC_FSPR | 7:0 | FSTP7 | FSTP6 | FSTP5 | FSTP4 | FSTP3 | FSTP2 | FSTP1 | FSTP0 |
| | | 15:8 | | | | | | FSTP10 | FSTP9 | FSTP8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x78 | PMC_FOCR | 7:0 | | | | | | | | FOCLR |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x7C ... 0x7F | Reserved | | | | | | | | | |
| 0x80 | PMC_PLLICPR | 7:0 | | | | | | | ICP_PLLA[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | ICP_PLLU[1:0] | |
| | | 31:24 | | | | | | | IVCO_PLLU[1:0] | |
| 0x84 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | PMC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | PMC_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xEC ... 0xFF | Reserved | | | | | | | | | |
| 0x0100 | PMC_PCER1 | 7:0 | PID39 | PID38 | PID37 | PID36 | PID35 | PID34 | PID33 | PID32 |
| | | 15:8 | PID47 | PID46 | PID45 | PID44 | PID43 | PID42 | PID41 | PID40 |
| | | 23:16 | PID55 | PID54 | PID53 | PID52 | PID51 | PID50 | PID49 | PID48 |
| | | 31:24 | PID63 | PID62 | PID61 | PID60 | PID59 | PID58 | PID57 | PID56 |
| 0x0104 | PMC_PCDR1 | 7:0 | PID39 | PID38 | PID37 | PID36 | PID35 | PID34 | PID33 | PID32 |
| | | 15:8 | PID47 | PID46 | PID45 | PID44 | PID43 | PID42 | PID41 | PID40 |
| | | 23:16 | PID55 | PID54 | PID53 | PID52 | PID51 | PID50 | PID49 | PID48 |
| | | 31:24 | PID63 | PID62 | PID61 | PID60 | PID59 | PID58 | PID57 | PID56 |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0108 | PMC_PCSR1 | 7:0 | PID39 | PID38 | PID37 | PID36 | PID35 | PID34 | PID33 | PID32 |
| | | 15:8 | PID47 | PID46 | PID45 | PID44 | PID43 | PID42 | PID41 | PID40 |
| | | 23:16 | PID55 | PID54 | PID53 | PID52 | PID51 | PID50 | PID49 | PID48 |
| | | 31:24 | PID63 | PID62 | PID61 | PID60 | PID59 | PID58 | PID57 | PID56 |
| 0x010C | PMC_PCR | 7:0 | | PID[6:0] | | | | | | |
| | | 15:8 | | | | CMD | | GCKCSS[2:0] | | |
| | | 23:16 | GCKDIV[3:0] | | | | | | | |
| | | 31:24 | | | GCKEN | EN | GCKDIV[7:4] | | | |
| 0x0110 | PMC_OCR | 7:0 | SEL | CAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0114 | PMC_SLPWK_ER0 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | PID23 | PID22 | PID21 | PID20 | PID19 | | | |
| | | 31:24 | | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| 0x0118 | PMC_SLPWK_DR0 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | PID23 | PID22 | PID21 | PID20 | PID19 | | | |
| | | 31:24 | | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| 0x011C | PMC_SLPWK_SR0 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | PID23 | PID22 | PID21 | PID20 | PID19 | | | |
| | | 31:24 | | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| 0x0120 | PMC_SLPWK_ASR0 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | PID23 | PID22 | PID21 | PID20 | PID19 | | | |
| | | 31:24 | | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| 0x0124 ... 0x0133 | Reserved | | | | | | | | | |
| 0x0134 | PMC_SLPWK_ER1 | 7:0 | | | | | | PID34 | PID33 | |
| | | 15:8 | | | | | | | | PID40 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0138 | PMC_SLPWK_DR1 | 7:0 | | | | | | PID34 | PID33 | |
| | | 15:8 | | | | | | | | PID40 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x013C | PMC_SLPWK_SR1 | 7:0 | | | | | | PID34 | PID33 | |
| | | 15:8 | | | | | | | | PID40 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0140 | PMC_SLPWK_ASR1 | 7:0 | | | | | | PID34 | PID33 | |
| | | 15:8 | | | | | | | | PID40 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0144 | PMC_SLPWK_AIPR | 7:0 | | | | | | | | AIP |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0148 | PMC_SLPWKCR | 7:0 | | PID[6:0] | | | | | | |
| | | 15:8 | | | | CMD | | | | |
| | | 23:16 | | | | | | | | ASR |
| | | 31:24 | | | | SLPWKSR | | | | |
| 0x014C | PMC_AUDIO_PLL0 | 7:0 | PLLFLT[3:0] | | | | RESETN | PMCEN | PADEN | PLLEN |
| | | 15:8 | | ND[6:0] | | | | | | |
| | | 23:16 | | QDPMC[6:0] | | | | | | |
| | | 31:24 | | | DCO_GAIN[1:0] | | DCO_FILTER[3:0] | | | |
| 0x0150 | PMC_AUDIO_PLL1 | 7:0 | FRACR[7:0] | | | | | | | |
| | | 15:8 | FRACR[15:8] | | | | | | | |
| | | 23:16 | | | FRACR[21:16] | | | | | |
| | | 31:24 | | QDAUDIO[4:0] | | | | | DIV[1:0] | |

### 32.22.1 PMC System Clock Enable Register

**Name:** PMC_SCER
**Offset:** 0x0000
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | ISCCK | | |
| Access | | | | | | W | | |
| Reset | | | | | | – | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PCK2 | PCK1 | PCK0 |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UDP | UHP | | | LCDCK | DDRCK | | |
| Access | W | W | | | W | W | | |
| Reset | – | – | | | – | – | | |

**Bit 18 – ISCCK** ISC Clock Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the ISC clock. |

**Bits 8, 9, 10 – PCKx** Programmable Clock x Output Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding Programmable Clock output. |

**Bit 7 – UDP** USB Device Clock Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the USB Device clock. |

**Bit 6 – UHP** USB Host OHCI Clocks Enable

| Value | Description |
|---|---|
| 0 | No effect. |

| Value | Description |
|---|---|
| 1 | Enables the UHP48M and UHP12M OHCI clocks. |

**Bit 3 – LCDCK** MCK2x Clock Enable
MCK2x is selected as LCD Pixel source clock if LCDC_LCDCFG0.CLKSEL = 1.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the MCK2x clock. |

**Bit 2 – DDRCK** DDR Clock Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the DDR clock. |

### 32.22.2 PMC System Clock Disable Register

**Name:** PMC_SCDR
**Offset:** 0x0004
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | ISCCK | | |
| Access | | | | | | W | | |
| Reset | | | | | | – | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PCK2 | PCK1 | PCK0 |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UDP | UHP | | | LCDCK | DDRCK | | PCK |
| Access | W | W | | | W | W | | W |
| Reset | – | – | | | – | – | | – |

**Bit 18 – ISCCK** ISC Clock Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the ISC clock. |

**Bits 8, 9, 10 – PCKx** Programmable Clock x Output Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding Programmable Clock output. |

**Bit 7 – UDP** USB Device Clock Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the USB Device clock. |

**Bit 6 – UHP** USB Host OHCI Clock Disable

| Value | Description |
|---|---|
| 0 | No effect. |

| Value | Description |
|---|---|
| 1 | Disables the UHP48M and UHP12M OHCI clocks. |

**Bit 3 – LCDCK** MCK2x Clock Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the MCK2x clock. |

**Bit 2 – DDRCK** DDR Clock Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the DDR clock. |

**Bit 0 – PCK** Processor Clock Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Processor clock. This is used to enter the processor in Idle mode. |

### 32.22.3 PMC System Clock Status Register

**Name:** PMC_SCSR
**Offset:** 0x0008
**Reset:** 0x00000005
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | ISCCK | | |
| Access | | | | | | R | | |
| Reset | | | | | | 0 | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PCK2 | PCK1 | PCK0 |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UDP | UHP | | | LCDCK | DDRCK | | PCK |
| Access | R | R | | | R | R | | R |
| Reset | 0 | 0 | | | 0 | 1 | | 1 |

**Bit 18 – ISCCK** ISC Clock Status

| Value | Description |
|---|---|
| 0 | The ISC clock is disabled. |
| 1 | The ISC clock is enabled. |

**Bits 8, 9, 10 – PCKx** Programmable Clock x Output Status

| Value | Description |
|---|---|
| 0 | The corresponding Programmable Clock output is disabled. |
| 1 | The corresponding Programmable Clock output is enabled. |

**Bit 7 – UDP** USB Device Port Clock Status

| Value | Description |
|---|---|
| 0 | The USB Device clock is disabled. |
| 1 | The USB Device clock is enabled. |

**Bit 6 – UHP** USB Host Port Clock Status

| Value | Description |
|---|---|
| 0 | The UHP48M and UHP12M OHCI clocks are disabled. |
| 1 | The UHP48M and UHP12M OHCI clocks are enabled. |

**Bit 3 – LCDCK** MCK2x Clock Status
MCK2x is selected as LCD Pixel source clock if LCDC_LCDCFG0.CLKSEL = 1.

| Value | Description |
|---|---|
| 0 | The MCK2x clock is disabled. |
| 1 | The MCK2x clock is enabled. |

**Bit 2 – DDRCK** DDR Clock Status

| Value | Description |
|---|---|
| 0 | The DDR clock is disabled. |
| 1 | The DDR clock is enabled. |

**Bit 0 – PCK** Processor Clock Status

| Value | Description |
|---|---|
| 0 | The Processor clock is disabled. |
| 1 | The Processor clock is enabled. |

### 32.22.4 PMC Peripheral Clock Enable Register 0

**Name:** PMC_PCER0
**Offset:** 0x0010
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID7 | PID6 | PID5 | PID4 | PID3 | PID2 | | |
| Access | W | W | W | W | W | W | | |
| Reset | – | – | – | – | – | – | | |

**Bits 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – PIDx** Peripheral Clock x Enable
PID2 to PID31 refer to identifiers as defined in section “Peripheral Identifiers”. Other peripherals can be enabled in PMC_PCER1.
Programming the control bits of the Peripheral ID that are not implemented has no effect on the behavior of the PMC.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding peripheral clock. |

### 32.22.5 PMC Peripheral Clock Disable Register 0

**Name:** PMC_PCDR0
**Offset:** 0x0014
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID7 | PID6 | PID5 | PID4 | PID3 | PID2 | | |
| Access | W | W | W | W | W | W | | |
| Reset | – | – | – | – | – | – | | |

**Bits 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – PIDx** Peripheral Clock x Disable
PID2 to PID31 refer to identifiers as defined in section “Peripheral Identifiers”. Other peripherals can be disabled in PMC_PCDR1.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding peripheral clock. |

### 32.22.6 PMC Peripheral Clock Status Register 0

**Name:** PMC_PCSR0
**Offset:** 0x0018
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID7 | PID6 | PID5 | PID4 | PID3 | PID2 | | |
| Access | R | R | R | R | R | R | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – PIDx** Peripheral Clock x Status
PID2 to PID31 refer to identifiers as defined in section “Peripheral Identifiers”. Other peripherals status can be read in PMC_PCSR1.

| Value | Description |
|---|---|
| 0 | The corresponding peripheral clock is disabled. |
| 1 | The corresponding peripheral clock is enabled. |

### 32.22.7 PMC UTMI Clock Configuration Register

**Name:** CKGR_UCKR
**Offset:** 0x001C
**Reset:** 0x10200000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BIASCOUNT[3:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | | | | |
| Reset | 0 | 0 | 0 | 1 | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UPLLCOUNT[3:0] | | | | | | | UPLLEN |
| Access | R/W | R/W | R/W | R/W | | | | R/W |
| Reset | 0 | 0 | 1 | 0 | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 31:28 – BIASCOUNT[3:0]** UTMI BIAS Start-up Time
Specifies the number of slow clock cycles for the UTMI BIAS start-up time.

**Bits 23:20 – UPLLCOUNT[3:0]** UTMI PLL Start-up Time
Specifies the number of slow clock cycles multiplied by 8 for the UTMI PLL start-up time.

**Bit 16 – UPLLEN** UTMI PLL Enable
When UPLLEN is set, the LOCKU flag is set once the UTMI PLL start-up time is achieved.

| Value | Description |
|---|---|
| 0 | The UTMI PLL is disabled. |
| 1 | The UTMI PLL is enabled. |

### 32.22.8 PMC Clock Generator Main Oscillator Register

**Name:** CKGR_MOR
**Offset:** 0x0020
**Reset:** 0x01000021
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CFDEN | MOSCSEL |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MOSCXTST[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | ONE | | MOSCRCEN | WAITMODE | MOSCXTBY | MOSCXTEN |
| Access | | | R/W | | R/W | R/W | R/W | R/W |
| Reset | | | 1 | | 0 | 0 | 0 | 1 |

**Bit 25 – CFDEN** Clock Failure Detector Enable

| Value | Description |
|---|---|
| 0 | The clock failure detector is disabled. |
| 1 | The clock failure detector is enabled. |

**Bit 24 – MOSCSEL** Main Clock Oscillator Selection

| Value | Description |
|---|---|
| 0 | The 12 MHz RC oscillator is selected. |
| 1 | The 8 to 24 MHz crystal oscillator is selected. |

**Bits 23:16 – KEY[7:0]** Password

| Value | Name | Description |
|---|---|---|
| 0x37 | PASSWD | Writing any other value in this field aborts the write operation. |

**Bits 15:8 – MOSCXTST[7:0]** 8 to 24 MHz Crystal Oscillator Start-up Time
Specifies the number of Slow clock cycles multiplied by 8 for the crystal oscillator start-up time.

**Bit 5 – ONE** Must Be Set to 1
When programming CKGR_MOR, bit 5 must always be set to 1; bits 6 and 4 must always be set to 0.

**Bit 3 – MOSCRCEN** 12 MHz RC Oscillator Enable
When MOSCRCEN is set, the MOSCRCS flag is set once the RC oscillator start-up time is achieved.

| Value | Description |
|---|---|
| 0 | The 12 MHz RC oscillator is disabled. |
| 1 | The 12 MHz RC oscillator is enabled. |

**Bit 2 – WAITMODE** Wait Mode Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Puts the device in Wait mode. |

**Bit 1 – MOSCXTBY** 8 to 24 MHz Crystal Oscillator Bypass
When MOSCXTBY is set, the MOSCXTS flag in PMC_SR is automatically set.

Clearing MOSCXTEN and MOSCXTBY bits allows resetting the MOSCXTS flag.

When Main Oscillator Bypass is disabled (MOSCXTBY = 0), the MOSCXTS flag must be read as 0 in PMC_SR prior to enabling the main crystal oscillator (MOSCXTEN = 1).

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The 8 to 24 MHz crystal oscillator is bypassed. MOSCXTEN must be cleared. An external clock must be connected on XIN. |

**Bit 0 – MOSCXTEN** 8 to 24 MHz Crystal Oscillator Enable
A crystal must be connected between XIN and XOUT.

When MOSCXTEN is set, the MOSCXTS flag is set once the crystal oscillator startup time is achieved.

| Value | Description |
|---|---|
| 0 | The 8 to 24 MHz crystal oscillator is disabled. |
| 1 | The 8 to 24 MHz crystal oscillator is enabled. MOSCXTBY must be cleared. |

Bits 4 and 6 must always be configured to 1.

### 32.22.9 PMC Clock Generator Main Clock Frequency Register

**Name:** CKGR_MCFR
**Offset:** 0x0024
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CCSS |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | RCMEAS | | | | MAINFRDY |
| Access | | | | R/W | | | | R/W |
| Reset | | | | 0 | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MAINF[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MAINF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 24 – CCSS** Counter Clock Source Selection

| Value | Description |
|---|---|
| 0 | The clock of the MAINF counter is the RC oscillator. |
| 1 | The clock of the MAINF counter is the crystal oscillator. |

**Bit 20 – RCMEAS** RC Oscillator Frequency Measure (write-only)
The measure is performed on the main frequency (i.e., not limited to RC oscillator only), but if the Main clock frequency source is the 8 to 24 MHz crystal oscillator, the restart of measuring is not needed because of the well known stability of crystal oscillators.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Restarts measuring of the frequency of the Main clock source. MAINF will carry the new frequency as soon as a low to high transition occurs on the MAINFRDY flag. |

**Bit 16 – MAINFRDY** Main Clock Frequency Measure Ready
To ensure that a correct value is read on the MAINF field, the MAINFRDY flag must be read at 1, then another read access must be performed on the register to get a stable value on the MAINF field.

| Value | Description |
|---|---|
| 0 | MAINF value is not valid or the measured oscillator is disabled or a measure has just been started by means of RCMEAS. |
| 1 | The measured oscillator has been enabled previously and MAINF value is available. |

**Bits 15:0 – MAINF[15:0]** Main Clock Frequency
Gives the number of cycles of the clock selected by the bit CCSS within 16 Slow clock periods. To calculate the frequency of the measured clock:

$f_{SELCK} = (MAINF \times f_{SLCK}) / 16$

where frequency is in MHz.

### 32.22.10 PMC Clock Generator PLLA Register

**Name:** CKGR_PLLAR
**Offset:** 0x0028
**Reset:** 0x00003F00
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

Possible limitations on PLL input frequencies and multiplier factors should be checked before using the PMC.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | ONE | | | | | MULA[6:6] |
| Access | | | R/W | | | | | R/W |
| Reset | | | 0 | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MULA[5:0] | | | | | | OUTA[3:2] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | OUTA[1:0] | | PLLACOUNT[5:0] | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DIVA |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 29 – ONE** Must Be Set to 1
Bit 29 must always be set to 1 when programming CKGR_PLLAR.

**Bits 24:18 – MULA[6:0]** PLLA Multiplier

| Value | Description |
|---|---|
| 0 | The PLLA is disabled. |
| 1-127 | The PLLA Clock frequency is the PLLA input frequency multiplied by MULA + 1. |

**Bits 17:14 – OUTA[3:0]** PLLA Clock Frequency Range
To be programmed to 0.

**Bits 13:8 – PLLACOUNT[5:0]** PLLA Counter
Specifies the number of Slow clock cycles before the LOCKA bit is set in PMC_SR after CKGR_PLLAR is written.

**Bit 0 – DIVA** Divider A

| Value | Name | Description |
|---|---|---|
| 0 | 0 | PLLA is disabled. |
| 1 | BYPASS | Divider is bypassed and the PLL input entry is Main clock (MAINCK). |

### 32.22.11 PMC Master Clock Register

**Name:** PMC_MCKR
**Offset:** 0x0030
**Reset:** 0x00000001
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | H32MXDIV |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | PLLADIV2 | | | MDIV[1:0] | |
| Access | | | | R/W | | | R/W | R/W |
| Reset | | | | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | PRES[2:0] | | | | | CSS[1:0] | |
| Access | | R/W | R/W | R/W | | | R/W | R/W |
| Reset | | 0 | 0 | 0 | | | 0 | 1 |

**Bit 24 – H32MXDIV** AHB 32-bit Matrix Divisor

| Value | Name | Description |
|---|---|---|
| 0 | H32MXDIV1 | The AHB 32-bit Matrix frequency is equal to the AHB 64-bit Matrix frequency. It is possible only if the AHB 64-bit Matrix frequency does not exceed 83 MHz. |
| 1 | H32MXDIV2 | The AHB 32-bit Matrix frequency is equal to the AHB 64-bit Matrix frequency divided by 2. |

**Bit 12 – PLLADIV2** PLLA Divisor by 2
Bit PLLADIV2 must always be set to 1 when MDIV is set to 3.

**Bits 9:8 – MDIV[1:0]** Master Clock Division

| Value | Name | Description |
|---|---|---|
| 0 | EQ_PCK | Master Clock is Prescaler Output Clock divided by 1.<br>**Warning: DDRCK is not available.** |
| 1 | PCK_DIV2 | Master Clock is Prescaler Output Clock divided by 2. DDRCK is equal to MCK. |
| 2 | PCK_DIV4 | Master Clock is Prescaler Output Clock divided by 4. DDRCK is equal to MCK. |
| 3 | PCK_DIV3 | Master Clock is Prescaler Output Clock divided by 3. DDRCK is equal to MCK. |

**Bits 6:4 – PRES[2:0]** Master/Processor Clock Prescaler

| Value | Name | Description |
|---|---|---|
| 0 | CLOCK | Selected clock |
| 1 | CLOCK_DIV2 | Selected clock divided by 2 |
| 2 | CLOCK_DIV4 | Selected clock divided by 4 |
| 3 | CLOCK_DIV8 | Selected clock divided by 8 |
| 4 | CLOCK_DIV16 | Selected clock divided by 16 |
| 5 | CLOCK_DIV32 | Selected clock divided by 32 |
| 6 | CLOCK_DIV64 | Selected clock divided by 64 |
| 7 | – | Reserved |

**Bits 1:0 – CSS[1:0]** Master/Processor Clock Source Selection

| Value | Name | Description |
|---|---|---|
| 0 | SLOW_CLK | Slow clock is selected |
| 1 | MAIN_CLK | Main clock is selected |
| 2 | PLLA_CLK | PLLACK is selected |
| 3 | UPLL_CLK | UPLL Clock is selected |

### 32.22.12 PMC USB Clock Register

**Name:** PMC_USB
**Offset:** 0x0038
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | USBDIV[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | USBS |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 11:8 – USBDIV[3:0]** Divider for USB OHCI Clock
USB Clock is Input clock divided by USBDIV + 1.

**Bit 0 – USBS** USB OHCI Input Clock Selection

| Value | Description |
|---|---|
| 0 | USB Clock Input is PLLA. |
| 1 | USB Clock Input is UPLL. |

### 32.22.13 PMC Programmable Clock Register

**Name:** PMC_PCKx
**Offset:** 0x40 + x*0x04 [x=0..2]
**Reset:** 0x00000000
**Property:** R/W

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PRES[7:4] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PRES[3:0] | | | | | CSS[2:0] | | |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bits 11:4 – PRES[7:0]** Programmable Clock Prescaler
Programmable Clock Frequency = Selected Clock Frequency / (PRES + 1)

| Value | Name | Description |
|---|---|---|
| 0 | CLOCK | Selected clock |
| 1 | CLOCK_DIV2 | Selected clock divided by 2 |
| 2 | CLOCK_DIV4 | Selected clock divided by 4 |
| 3 | CLOCK_DIV8 | Selected clock divided by 8 |
| 4 | CLOCK_DIV16 | Selected clock divided by 16 |
| 5 | CLOCK_DIV32 | Selected clock divided by 32 |
| 6 | CLOCK_DIV64 | Selected clock divided by 64 |
| 7 | – | Reserved |

**Bits 2:0 – CSS[2:0]** Master Clock Source Selection

| Value | Name | Description |
|---|---|---|
| 0 | SLOW_CLK | Slow clock is selected. |
| 1 | MAIN_CLK | Main clock is selected. |
| 2 | PLLA_CLK | PLLACK is selected. |
| 3 | UPLL_CLK | UPLL clock is selected. |

| Value | Name | Description |
|---|---|---|
| 4 | MCK_CLK | Master clock is selected. |
| 5 | AUDIO_CLK | Audio PLL clock is selected. |

### 32.22.14 PMC Interrupt Enable Register

**Name:** PMC_IER
**Offset:** 0x0060
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CFDEV | MOSCRCS | MOSCSELS |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PCKRDYx[2:0] | | |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | LOCKU | | | MCKRDY | | LOCKA | MOSCXTS |
| Access | | W | | | W | | W | W |
| Reset | | – | | | – | | – | – |

**Bit 18 – CFDEV** Clock Failure Detector Event Interrupt Enable

**Bit 17 – MOSCRCS** 12 MHz RC Oscillator Status Interrupt Enable

**Bit 16 – MOSCSELS** Main Clock Source Oscillator Selection Status Interrupt Enable

**Bits 10:8 – PCKRDYx[2:0]** Programmable Clock Ready x Interrupt Enable

**Bit 6 – LOCKU** UTMI PLL Lock Interrupt Enable

**Bit 3 – MCKRDY** Master Clock Ready Interrupt Enable

**Bit 1 – LOCKA** PLLA Lock Interrupt Enable

**Bit 0 – MOSCXTS** 8 to 24 MHz Crystal Oscillator Status Interrupt Enable

### 32.22.15 PMC Interrupt Disable Register

**Name:** PMC_IDR
**Offset:** 0x0064
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CFDEV | MOSCRCS | MOSCSELS |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PCKRDYx[2:0] | | |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | LOCKU | | | MCKRDY | | LOCKA | MOSCXTS |
| Access | | W | | | W | | W | W |
| Reset | | – | | | – | | – | – |

**Bit 18 – CFDEV** Clock Failure Detector Event Interrupt Disable

**Bit 17 – MOSCRCS** 12 MHz RC Oscillator Status Interrupt Disable

**Bit 16 – MOSCSELS** Main Oscillator Clock Source Selection Status Interrupt Disable

**Bits 10:8 – PCKRDYx[2:0]** Programmable Clock Ready x Interrupt Disable

**Bit 6 – LOCKU** UTMI PLL Lock Interrupt Enable

**Bit 3 – MCKRDY** Master Clock Ready Interrupt Disable

**Bit 1 – LOCKA** PLLA Lock Interrupt Disable

**Bit 0 – MOSCXTS** 8 to 24 MHz Crystal Oscillator Status Interrupt Disable

### 32.22.16 PMC Status Register

**Name:** PMC_SR
**Offset:** 0x0068
**Reset:** 0x00010008
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | GCKRDY |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | FOS | CFDS | CFDEV | MOSCRCS | MOSCSELS |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PCKRDYx[2:0] | | |
| Access | | | | | | | | |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OSCSELS | LOCKU | | | MCKRDY | | LOCKA | MOSCXTS |
| Access | R | R | | | R | | R | R |
| Reset | 0 | 0 | | | 1 | | 0 | 0 |

**Bit 24 – GCKRDY** Generic Clock Status

| Value | Description |
|---|---|
| 0 | One of the generic clocks is not ready yet. |
| 1 | All generic clocks are ready. |

**Bit 20 – FOS** Clock Failure Detector Fault Output Status

| Value | Description |
|---|---|
| 0 | The fault output of the clock failure detector is inactive. |
| 1 | The fault output of the clock failure detector is active. |

**Bit 19 – CFDS** Clock Failure Detector Status

| Value | Description |
|---|---|
| 0 | A clock failure of the 8 to 24 MHz crystal oscillator is not detected. |
| 1 | A clock failure of the 8 to 24 MHz crystal oscillator is detected. |

**Bit 18 – CFDEV** Clock Failure Detector Event

| Value | Description |
|---|---|
| 0 | No clock failure detection of the 8 to 24 MHz crystal oscillator has occurred since the last read of PMC_SR. |

| Value | Description |
|---|---|
| 1 | At least one clock failure detection of the 8 to 24 MHz crystal oscillator has occurred since the last read of PMC_SR. |

**Bit 17 – MOSCRCS** 12 MHz RC Oscillator Status

| Value | Description |
|---|---|
| 0 | 12 MHz RC oscillator is not stabilized. |
| 1 | 12 MHz RC oscillator is stabilized. |

**Bit 16 – MOSCSELS** Main Oscillator Selection Status

| Value | Description |
|---|---|
| 0 | Selection is in progress. |
| 1 | Selection is done. |

**Bits 10:8 – PCKRDYx[2:0]** Programmable Clock Ready Status

| Value | Description |
|---|---|
| 0 | Programmable Clock x is not ready. |
| 1 | Programmable Clock x is ready. |

**Bit 7 – OSCSELS** Slow Clock Oscillator Selection

| Value | Description |
|---|---|
| 0 | Embedded 64 kHz RC oscillator is selected. |
| 1 | 32.768 kHz crystal oscillator is selected. |

**Bit 6 – LOCKU** UPLL Clock Status

| Value | Description |
|---|---|
| 0 | UPLL Clock is not ready. |
| 1 | UPLL Clock is ready. |

**Bit 3 – MCKRDY** Master Clock Status

| Value | Description |
|---|---|
| 0 | Master Clock is not ready. |
| 1 | Master Clock is ready. |

**Bit 1 – LOCKA** PLLA Lock Status

| Value | Description |
|---|---|
| 0 | PLLA is not locked. |
| 1 | PLLA is locked. |

**Bit 0 – MOSCXTS** 8 to 24 MHz Crystal Oscillator Status

| Value | Description |
|---|---|
| 0 | 8 to 24 MHz crystal oscillator is not stabilized. |
| 1 | 8 to 24 MHz crystal oscillator is stabilized. |

### 32.22.17 PMC Interrupt Mask Register

**Name:** PMC_IMR
**Offset:** 0x006C
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: Corresponding interrupt is not enabled.

1: Corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CFDEV | MOSCRCS | MOSCSELS |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PCKRDYx[2:0] | | |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MCKRDY | | LOCKA | MOSCXTS |
| Access | | | | | R | | R | R |
| Reset | | | | | 0 | | 0 | 0 |

**Bit 18 – CFDEV** Clock Failure Detector Event Interrupt Mask

**Bit 17 – MOSCRCS** 12 MHz RC Oscillator Status Interrupt Mask

**Bit 16 – MOSCSELS** Main Oscillator Clock Source Selection Status Interrupt Mask

**Bits 10:8 – PCKRDYx[2:0]** Programmable Clock Ready x Interrupt Mask

**Bit 3 – MCKRDY** Master Clock Ready Interrupt Mask

**Bit 1 – LOCKA** PLLA Lock Interrupt Mask

**Bit 0 – MOSCXTS** 8 to 24 MHz Crystal Oscillator Status Interrupt Mask

### 32.22.18 PMC Fast Startup Polarity Register

**Name:** PMC_FSPR
**Offset:** 0x0074
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | FSTP10 | FSTP9 | FSTP8 |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FSTP7 | FSTP6 | FSTP5 | FSTP4 | FSTP3 | FSTP2 | FSTP1 | FSTP0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 10 – FSTP10** GMAC Wake-up On LAN Polarity for Fast Start-up
If PMC_FSMR.FSTT10 = 1, FSTP10 must be written to 1.

**Bits 2, 3, 4, 5, 6, 7, 8, 9 – FSTPx** PIOBU0–7 Pin Polarity for Fast Start-up
Defines the active polarity of the corresponding PIOBUx input. If the corresponding wake-up input is enabled and at the FSTP level, it enables a fast restart signal.

**Bit 1 – FSTP1** Security Module Polarity for Fast Start-up
If PMC_FSMR.FSTT1 = 1, FSTP1 must be written to 1.

**Bit 0 – FSTP0** WKUP Pin Polarity for Fast Start-up
Defines the active polarity of the wake-up input. If the wake-up input is enabled and at the FSTP level, it enables a fast restart signal.

### 32.22.19 PMC Fast Start-up Mode Register

**Name:** PMC_FSMR
**Offset:** 0x0070
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ACC_CE | RXLP_MCE |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | LPM | SDMMC_CD | USBAL | RTCAL | |
| Access | | | | R/W | R/W | R/W | R/W | |
| Reset | | | | 0 | 0 | 0 | 0 | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | FSTT10 | FSTT9 | FSTT8 |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FSTT7 | FSTT6 | FSTT5 | FSTT4 | FSTT3 | FSTT2 | FSTT1 | FSTT0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 25 – ACC_CE** Fast Start-up from Analog Comparator Controller Comparison Enable

| Value | Description |
|---|---|
| 0 | The ACC (Analog Comparator Controller) comparison has no effect on the PMC. |
| 1 | The ACC (Analog Comparator Controller) comparison can trigger a fast restart signal to the PMC. |

**Bit 24 – RXLP_MCE** Fast Start-up from Backup UART Receive Match Condition Enable

| Value | Description |
|---|---|
| 0 | The matching condition on the RXLP has no effect on the PMC. |
| 1 | The matching condition on the RXLP can trigger a fast restart signal to the PMC. |

**Bit 20 – LPM** Low-power Mode

| Value | Description |
|---|---|
| 0 | The WaitForInterrupt (WFI) or the WaitForEvent (WFE) instruction of the processor instructs the processor to enter Idle mode. |
| 1 | The WaitForEvent (WFE) instruction of the processor instructs the system to enter ULP mode 1. |

**Bit 19 – SDMMC_CD** Fast Start-up from SDMMC Card Detect Enable

| Value | Description |
|---|---|
| 0 | The SDMMC card detect has no effect on the PMC. |
| 1 | The SDMMC card detect can trigger a fast restart signal to the PMC. |

**Bit 18 – USBAL** Fast Start-up from USB Resume Enable

| Value | Description |
|---|---|
| 0 | The USB resume has no effect on the PMC. |
| 1 | The USB resume can trigger a fast restart signal to the PMC. |

**Bit 17 – RTCAL** Fast Start-up from RTC Alarm Enable

| Value | Description |
|---|---|
| 0 | The RTC alarm has no effect on the PMC. |
| 1 | The RTC alarm can trigger a fast restart signal to the PMC. |

**Bit 10 – FSTT10** Fast Start-up from GMAC Wake-up On LAN Enable

| Value | Description |
|---|---|
| 0 | The GMAC_WOL input has no effect on the PMC. |
| 1 | The GMAC_WOL input can trigger a fast restart signal to the PMC. |

**Bits 2, 3, 4, 5, 6, 7, 8, 9 – FSTTx** Fast Start-up from PIOBU0–7 Input Enable

| Value | Description |
|---|---|
| 0 | The corresponding PIOBUx input has no effect on the PMC. |
| 1 | The corresponding PIOBUx input can trigger a fast restart signal to the PMC. |

**Bit 1 – FSTT1** Fast Start-up from Security Module Enable

| Value | Description |
|---|---|
| 0 | The SECUMOD has no effect on the PMC. |
| 1 | The SECUMOD can trigger a fast restart signal to the PMC. |

**Bit 0 – FSTT0** Fast Start-up from WKUP Pin Enable

| Value | Description |
|---|---|
| 0 | The wake-up input (WKUP) has no effect on the PMC. |
| 1 | The wake-up input (WKUP) can trigger a fast restart signal to the PMC. |

### 32.22.20 PMC Fault Output Clear Register

**Name:** PMC_FOCR
**Offset:** 0x0078
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | FOCLR |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – FOCLR** Fault Output Clear
Clears the clock failure detector fault output.

### 32.22.21 PLL Charge Pump Current Register

**Name:** PMC_PLLICPR
**Offset:** 0x0080
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | IVCO_PLLU[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ICP_PLLU[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ICP_PLLA[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bits 25:24 – IVCO_PLLU[1:0]** Voltage Control Output Current PLL UTMI
Should be written to 0.

**Bits 17:16 – ICP_PLLU[1:0]** Charge Pump Current PLL UTMI
Should be written to 0.

**Bits 1:0 – ICP_PLLA[1:0]** Charge Pump Current
To optimize clock performance, this field must be programmed as specified in “PLL A Characteristics” in the Electrical Characteristics section.

### 32.22.22 PMC Write Protection Mode Register

**Name:** PMC_WPMR
**Offset:** 0x00E4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x504D43 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit.<br>Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See Register Write Protection for the list of registers that can be write-protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x504D43 (“PMC” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x504D43 (“PMC” in ASCII). |

### 32.22.23 PMC Write Protection Status Register

**Name:** PMC_WPSR
**Offset:** 0x00E8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of PMC_WPSR. |
| 1 | A write protection violation has occurred since the last read of PMC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

### 32.22.24 PMC Peripheral Clock Enable Register 1

**Name:** PMC_PCER1
**Offset:** 0x0100
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID63 | PID62 | PID61 | PID60 | PID59 | PID58 | PID57 | PID56 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID55 | PID54 | PID53 | PID52 | PID51 | PID50 | PID49 | PID48 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID47 | PID46 | PID45 | PID44 | PID43 | PID42 | PID41 | PID40 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID39 | PID38 | PID37 | PID36 | PID35 | PID34 | PID33 | PID32 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – PIDx** Peripheral Clock x Enable
PID32 to PID63 refer to identifiers as defined in the section “Peripheral Identifiers”. Programming the control bits of the Peripheral ID that are not implemented has no effect on the behavior of the PMC.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding peripheral clock. |

### 32.22.25 PMC Peripheral Clock Disable Register 1

**Name:** PMC_PCDR1
**Offset:** 0x0104
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID63 | PID62 | PID61 | PID60 | PID59 | PID58 | PID57 | PID56 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID55 | PID54 | PID53 | PID52 | PID51 | PID50 | PID49 | PID48 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID47 | PID46 | PID45 | PID44 | PID43 | PID42 | PID41 | PID40 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID39 | PID38 | PID37 | PID36 | PID35 | PID34 | PID33 | PID32 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – PIDx** Peripheral Clock x Disable
PID32 to PID63 refer to identifiers as defined in the section “Peripheral Identifiers”.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding peripheral clock. |

### 32.22.26 PMC Peripheral Clock Status Register 1

**Name:** PMC_PCSR1
**Offset:** 0x0108
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID63 | PID62 | PID61 | PID60 | PID59 | PID58 | PID57 | PID56 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID55 | PID54 | PID53 | PID52 | PID51 | PID50 | PID49 | PID48 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID47 | PID46 | PID45 | PID44 | PID43 | PID42 | PID41 | PID40 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID39 | PID38 | PID37 | PID36 | PID35 | PID34 | PID33 | PID32 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – PIDx** Peripheral Clock x Status
PID32 to PID63 refer to identifiers as defined in the section “Peripheral Identifiers”.

| Value | Description |
|---|---|
| 0 | The corresponding peripheral clock is disabled. |
| 1 | The corresponding peripheral clock is enabled. |

### 32.22.27 PMC Peripheral Control Register

**Name:** PMC_PCR
**Offset:** 0x010C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | GCKEN | EN | GCKDIV[7:4] | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GCKDIV[3:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | | | | |
| Reset | 0 | 0 | 0 | 0 | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | CMD | | GCKCSS[2:0] | | |
| Access | | | | R/W | | R/W | R/W | R/W |
| Reset | | | | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | PID[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 29 – GCKEN** Generic Clock Enable

| Value | Description |
|---|---|
| 0 | The selected generic clock is disabled. |
| 1 | The selected generic clock is enabled. |

**Bit 28 – EN** Enable

| Value | Description |
|---|---|
| 0 | The selected peripheral clock is disabled. |
| 1 | The selected peripheral clock is enabled. |

**Bits 27:20 – GCKDIV[7:0]** Generic Clock Division Ratio
Generic clock is: selected clock period divided by GCKDIV + 1. GCKDIV must not be changed while the peripheral selects GCLK (e.g., bit rate, etc.).

**Bit 12 – CMD** Command

| Value | Description |
|---|---|
| 0 | Read mode |
| 1 | Write mode |

**Bits 10:8 – GCKCSS[2:0]** Generic Clock Source Selection

| Value | Name | Description |
|---|---|---|
| 0 | SLOW_CLK | Slow clock is selected. |
| 1 | MAIN_CLK | Main clock is selected. |
| 2 | PLLA_CLK | PLLACK is selected. |
| 3 | UPLL_CLK | UPLL Clock is selected. |
| 4 | MCK_CLK | Master Clock is selected. |
| 5 | AUDIO_CLK | Audio PLL clock is selected. |

**Bits 6:0 – PID[6:0]** Peripheral ID
Peripheral ID selection from PID2 to the maximum PID number. This refers to identifiers as defined in the section “Peripheral Identifiers”.

### 32.22.28 PMC Oscillator Calibration Register

**Name:** PMC_OCR
**Offset:** 0x0110
**Reset:** 0x00404040
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SEL | CAL[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – SEL** Selection of RC Oscillator Calibration Bits

| Value | Description |
|---|---|
| 0 | Factory determined value. |
| 1 | Value written by user in CAL field of this register. |

**Bits 6:0 – CAL[6:0]** 12 MHz RC Oscillator Calibration Bits
Calibration bits applied to the RC oscillator when SEL is set.

### 32.22.29 PMC SleepWalking Enable Register 0

**Name:** PMC_SLPWK_ER0
**Offset:** 0x0114
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| Access | | W | W | W | W | W | W | W |
| Reset | | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID23 | PID22 | PID21 | PID20 | PID19 | | | |
| Access | W | W | W | W | W | | | |
| Reset | – | – | – | – | – | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30 – PIDx** Peripheral x SleepWalking Enable
Not all PIDs can be configured with asynchronous partial wake-up.

Only the following PIDs can be configured with asynchronous partial wake-up: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

The clock of the peripheral must be enabled before using its asynchronous partial wake-up (SleepWalking) function (its associated PIDx field in ISCCK: ISC Clock Status or PMC Peripheral Clock Status Register 1 is set to ‘1’).

The values for PIDx are defined in section “Peripheral Identifiers”.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The asynchronous partial wake-up (SleepWalking) function of the corresponding peripheral is enabled. |

### 32.22.30 PMC SleepWalking Disable Register 0

**Name:** PMC_SLPWK_DR0
**Offset:** 0x0118
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| Access | | W | W | W | W | W | W | W |
| Reset | | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID23 | PID22 | PID21 | PID20 | PID19 | | | |
| Access | W | W | W | W | W | | | |
| Reset | – | – | – | – | – | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30 – PIDx** Peripheral x SleepWalking Disable
Not all PIDs can be configured with asynchronous partial wake-up.

Only the following PIDs can be configured with asynchronous partial wake-up: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

The values for PIDx are defined in the section “Peripheral Identifiers”.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The asynchronous partial wake-up (SleepWalking) function of the corresponding peripheral is disabled. |

### 32.22.31 PMC SleepWalking Status Register 0

**Name:** PMC_SLPWK_SR0
**Offset:** 0x011C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID23 | PID22 | PID21 | PID20 | PID19 | | | |
| Access | R | R | R | R | R | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30 – PIDx** Peripheral x SleepWalking Status
Not all PIDs can be configured with asynchronous partial wake-up.

Only the following PIDs can be configured with asynchronous partial wake-up: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

The values for PIDx are defined in the section “Peripheral Identifiers”.

| Value | Description |
|---|---|
| 0 | The asynchronous partial wake-up (SleepWalking) function of the peripheral is currently disabled or the peripheral enabled for asynchronous partial wake-up (SleepWalking) cleared the PIDx bit upon detection of a wake-up condition. |
| 1 | The asynchronous partial wake-up (SleepWalking) function of the peripheral is currently enabled. |

### 32.22.32 PMC SleepWalking Activity Status Register 0

**Name:** PMC_SLPWK_ASR0
**Offset:** 0x0120
**Reset:** –
**Property:** Read-Only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| Access | | R | R | R | R | R | R | R |
| Reset | | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID23 | PID22 | PID21 | PID20 | PID19 | | | |
| Access | R | R | R | R | R | | | |
| Reset | – | – | – | – | – | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30 – PIDx** Peripheral x Activity Status
Only the following PIDs can be configured with asynchronous partial wake-up: FLEXCOMx, SPIx, TWIx, UARTx and ADC.
All other PIDs are always read at 0.

The values for PIDx are defined in the section “Peripheral Identifiers”.

| Value | Description |
|---|---|
| 0 | The peripheral x is not presently active. The asynchronous partial wake-up (SleepWalking) function can be activated. |
| 1 | The peripheral x is presently active. The asynchronous partial wake-up (SleepWalking) function must not be activated. |

### 32.22.33 PMC SleepWalking Enable Register 1

**Name:** PMC_SLPWK_ER1
**Offset:** 0x0134
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

Not all PIDs can be configured with asynchronous partial wake-up.

Only the following PIDs can be configured with asynchronous partial wake-up: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

The clock of the peripheral must be enabled before using its asynchronous partial wake-up (SleepWalking) function (the associated PIDx field in PMC Peripheral Clock Status Register 1 or ISCCK: ISC Clock Status is set to ‘1’).

The values for PIDx are defined in the section “Peripheral Identifiers”.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: The asynchronous partial wake-up (SleepWalking) function of the corresponding peripheral is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | PID40 |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PID34 | PID33 | |
| Access | | | | | | W | W | |
| Reset | | | | | | – | – | |

**Bit 8 – PID40** Peripheral 40 SleepWalking Enable

**Bits 1, 2 – PIDx** Peripherals 33, 34 SleepWalking Enable

### 32.22.34 PMC SleepWalking Disable Register 1

**Name:** PMC_SLPWK_DR1
**Offset:** 0x0138
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

Not all PIDs can be configured with asynchronous partial wake-up.

Only the following PIDs can be configured with asynchronous partial wake-up: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

The values for PIDx are defined in the section “Peripheral Identifiers”.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: The asynchronous partial wake-up (SleepWalking) function of the corresponding peripheral is disabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | PID40 |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PID34 | PID33 | |
| Access | | | | | | W | W | |
| Reset | | | | | | – | – | |

**Bit 8 – PID40** Peripheral x SleepWalking Disable

**Bits 1, 2 – PIDx** Peripherals 33, 34 SleepWalking Disable

### 32.22.35 PMC SleepWalking Status Register 1

**Name:** PMC_SLPWK_SR1
**Offset:** 0x013C
**Reset:** 0x00000000
**Property:** Read-only

Not all PIDs can be configured with asynchronous partial wake-up.

Only the following PIDs can be configured with asynchronous partial wake-up: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

The values for PIDx are defined in the section “Peripheral Identifiers”.

The following configuration values are valid for all listed bit names of this register:

0: The asynchronous partial wake-up (SleepWalking) function of the peripheral is currently disabled or the peripheral enabled for asynchronous partial wake-up (SleepWalking) cleared the PIDx bit upon detection of a wake-up condition.

1: The asynchronous partial wake-up (SleepWalking) function of the peripheral is currently enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | PID40 |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PID34 | PID33 | |
| Access | | | | | | R | R | |
| Reset | | | | | | 0 | 0 | |

**Bit 8 – PID40** Peripheral 40 SleepWalking Status

**Bits 1, 2 – PIDx** Peripherals 33, 34 SleepWalking Status

### 32.22.36 PMC SleepWalking Activity Status Register 1

**Name:** PMC_SLPWK_ASR1
**Offset:** 0x0140
**Reset:** –
**Property:** Read-Only

Only the following PIDs can be configured with asynchronous partial wake-up: FLEXCOMx, SPIx, TWIx, UARTx and ADC.
All other PIDs are always read at 0.

The values for PIDx are defined in the section “Peripheral Identifiers”.

The following configuration values are valid for all listed bit names of this register:

0: The peripheral x is not currently active; the asynchronous partial wake-up (SleepWalking) function can be activated.

1: The peripheral x is currently active; the asynchronous partial wake-up (SleepWalking) function must not be activated.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | R |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | R | R | |
| Reset | | | | | | – | – | |

**Bit 8 – PID40** Peripheral 40 Activity Status

**Bits 1, 2 – PIDx** Peripherals 33, 34 Activity Status

### 32.22.37 PMC SleepWalking Activity In Progress Register

**Name:** PMC_SLPWK_AIPR
**Offset:** 0x0144
**Reset:** –
**Property:** Read-Only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | AIP |
| Access | | | | | | | | R |
| Reset | | | | | | | | – |

**Bit 0 – AIP** Activity In Progress
Only the following PIDs can be configured with asynchronous partial wake-up: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

| Value | Description |
|---|---|
| 0 | There is no activity on peripherals. The asynchronous partial wake-up (SleepWalking) function can be activated on one or more peripherals. The device can enter ULP mode 1. |
| 1 | One or more peripherals are currently active. The device must not enter ULP mode 1 if the asynchronous partial wake-up is enabled for one of the following PIDs: FLEXCOMx, SPIx, TWIx, UARTx and ADC. |

### 32.22.38 PMC SleepWalking Control Register

**Name:** PMC_SLPWKCR
**Offset:** 0x0148
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | SLPWKSR | | | | |
| Access | | | | R/W | | | | |
| Reset | | | | 0 | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ASR |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | CMD | | | | |
| Access | | | | R/W | | | | |
| Reset | | | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | PID[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 28 – SLPWKSR** SleepWalking Status Register
Not all PIDs can be configured with asynchronous partial wake-up.

Only the following PIDs can be configured with asynchronous partial wake-up: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

| Value | Description |
|---|---|
| 0 | The asynchronous partial wake-up (SleepWalking) function of the peripheral is disabled. |
| 1 | The asynchronous partial wake-up (SleepWalking) function of the peripheral is enabled. |

**Bit 16 – ASR** Activity Status Register
Not all PIDs can be configured with asynchronous partial wake-up.

Only the following PIDs can be configured with asynchronous partial wake-up: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

| Value | Description |
|---|---|
| 0 | The peripheral x is not currently active; the asynchronous partial wake-up (SleepWalking) function can be activated. |
| 1 | The peripheral x is currently active; the asynchronous partial wake-up (SleepWalking) function must not be activated. |

**Bit 12 – CMD** Command

| Value | Description |
|---|---|
| 0 | Read mode |
| 1 | Write mode |

**Bits 6:0 – PID[6:0]** Peripheral ID
Peripheral ID selection from PID2 to the maximum PID number. This refers to identifiers as defined in the section “Peripheral Identifiers”.

### 32.22.39 PMC Audio PLL Control Register 0

**Name:** PMC_AUDIO_PLL0
**Offset:** 0x014C
**Reset:** 0x000000D0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | DCO_GAIN[1:0] | | DCO_FILTER[3:0] | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | QDPMC[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | ND[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PLLFLT[3:0] | | | | RESETN | PMCEN | PADEN | PLLEN |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |

**Bits 29:28 – DCO_GAIN[1:0]** Digitally Controlled Oscillator Gain Selection
For optimization, the value of this field must be configured to 0.

**Bits 27:24 – DCO_FILTER[3:0]** Digitally Controlled Oscillator Filter Selection
For optimization, the value of this field must be configured to 0.

**Bits 22:16 – QDPMC[6:0]** Output Divider Ratio for PMC Clock
$f_{pmc} = f_{ref} \times ((ND + 1) + FRACR \div 2^{22}) / (QDPMC + 1)$

**Bits 14:8 – ND[6:0]** Loop Divider Ratio

**Bits 7:4 – PLLFLT[3:0]** PLL Loop Filter Selection
Default value should be 13 (0xD).

**Bit 3 – RESETN** Audio PLL Reset

| Value | Description |
|---|---|
| 0 | The audio PLL is in reset state. |
| 1 | The audio PLL is in active state. |

**Bit 2 – PMCEN** PMC Clock Enable

| Value | Description |
|---|---|
| 0 | The output clock of the audio PLL is not sent to the PMC. |
| 1 | The output clock of the audio PLL is sent to the PMC. |

**Bit 1 – PADEN** Pad Clock Enable

| Value | Description |
|---|---|
| 0 | The external audio pin CLK_AUDIO is driven low. |
| 1 | The external audio pin CLK_AUDIO is driven by AUDIOPINCLK. |

**Bit 0 – PLLEN** PLL Enable

| Value | Description |
|---|---|
| 0 | The Audio PLL is disabled. |
| 1 | The Audio PLL is enabled |

### 32.22.40 PMC Audio PLL Control Register 1

**Name:** PMC_AUDIO_PLL1
**Offset:** 0x0150
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | QDAUDIO[4:0] | | | | | DIV[1:0] | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | FRACR[21:16] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FRACR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FRACR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 30:26 – QDAUDIO[4:0]** Output Divider Ratio for Pad Clock
$f_{audio} = f_{ref} \times ((ND + 1) + FRACR \div 2^{22}) / (DIV \times QDAUDIO)$

**Bits 25:24 – DIV[1:0]** Divider Value

| Value | Name | Description |
|---|---|---|
| 0 | FORBIDDEN | Reserved |
| 1 | FORBIDDEN | Reserved |
| 2 | DIV2 | Divide by 2 |
| 3 | DIV3 | Divide by 3 |

**Bits 21:0 – FRACR[21:0]** Fractional Loop Divider Setting

# 33. Parallel Input/Output Controller (PIO)

## 33.1 Description

The Parallel Input/Output Controller (PIO) manages up to 128 fully programmable input/output lines. Each I/O line may be dedicated as a general purpose I/O or be assigned to a function of an embedded peripheral. This ensures effective optimization of the pins of the product.

Each I/O line of the PIO Controller features:

- An input change interrupt enabling level change detection on any I/O line
- Rising edge, falling edge, both edge, low-level or high-level detection on any I/O line
- A glitch filter providing rejection of glitches lower than one-half of PIO clock cycle
- A debouncing filter providing rejection of unwanted pulses from key or push button operations
- Multi-drive capability similar to an open drain I/O line
- Control of the pull-up and pull-down of the I/O line
- Input visibility and output control
- Secure or Non-Secure management of the I/O line

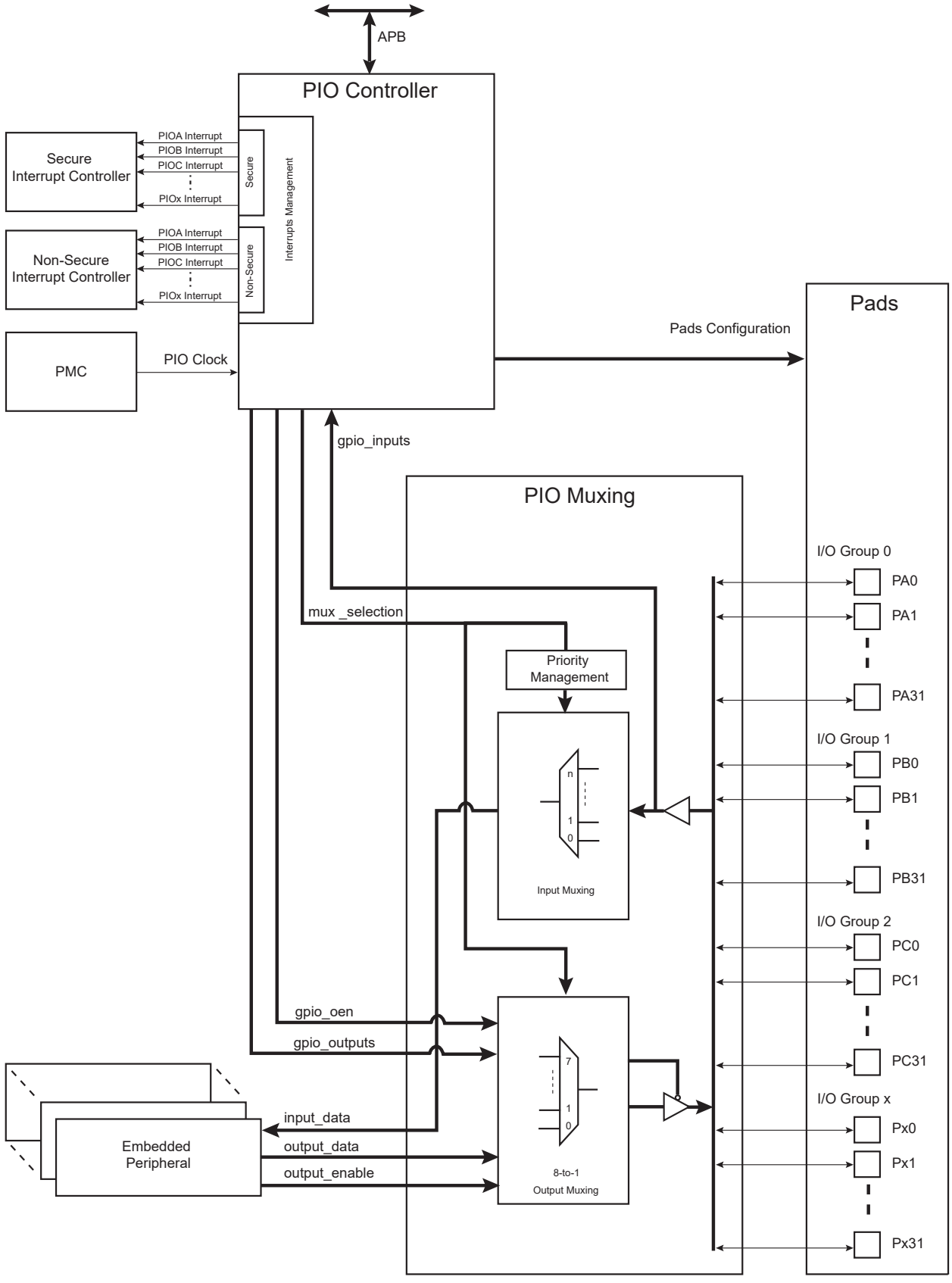
The PIO Controller also features a synchronous output providing up to 32 bits of data output in a single write operation.

## 33.2 Embedded Characteristics

- Up to 128 Programmable I/O Lines
- Multiplexing of up to 6 Peripheral Functions per I/O Line
- For each I/O Line (whether assigned to a peripheral or used as general purpose I/O):
  - Input Change Interrupt
  - Programmable Glitch Filter
  - Programmable Debouncing Filter
  - Multi-drive Option Enables Driving in Open Drain
  - Programmable Pull-Up/Pull-Down on Each I/O Line
  - Pin Data Status Register, Supplies Visibility of the Level on the Pin at Any Time
  - Programmable Event: Rising Edge, Falling Edge, Both edge, Low-Level or High-Level
  - Configuration Lock by the Connected Peripheral
  - Security management of each I/O line
  - Programmable Configuration Lock (Active Until Next $V_{DDCORE}$ Reset) to protect Against Further Software Modifications (intentional or unintentional)
- Register Write Protection against unintentional software modifications:
  - One configuration bit to enable or disable protection of I/O line settings
  - One configuration bit to enable or disable protection of interrupt settings
- Synchronous Output, possibility to set or clear simultaneously up to 32 I/O Lines in a Single Write
- Programmable Schmitt Trigger Inputs
- Programmable I/O Drive

## 33.3 Block Diagram

**Figure 33-1. Block Diagram**



**Note:**

1. x = 3 (the number of I/O groups is 4).
2. n depends on the number of I/O lines affected to the IP input.

## 33.4 Product Dependencies

### 33.4.1 Pin Multiplexing

Each pin is configurable, depending on the product, as either a general purpose I/O line only, or as an I/O line multiplexed with up to 6 peripheral I/Os. As the multiplexing is hardware defined and thus product-dependent, the hardware designer and programmer must carefully determine the configuration of the PIO Controllers required by their application. When an I/O line is general purpose only, i.e., not multiplexed with any peripheral I/O, programming of the PIO Controller regarding the assignment to a peripheral has no effect and only the PIO Controller can control how the pin is driven by the product.

### 33.4.2 External Interrupt Lines

The interrupt signals FIQ and IRQ0 to IRQn are multiplexed through the PIO Controllers.

### 33.4.3 Power Management

The Power Management Controller (PMC) controls the PIO Controller clock in order to save power. Writing any of the registers of the user interface does not require the PIO Controller clock to be enabled. This means that the configuration of the I/O lines does not require the PIO Controller clock to be enabled.

However, when the clock is disabled, not all of the features of the PIO Controller are available, including glitch filtering. Note that the input change interrupt, the interrupt modes on a programmable event and the read of the pin level require the clock to be validated.

After a hardware reset, the PIO clock is disabled by default.

The user must configure the PMC before any access to the input line information.

### 33.4.4 Interrupt Generation

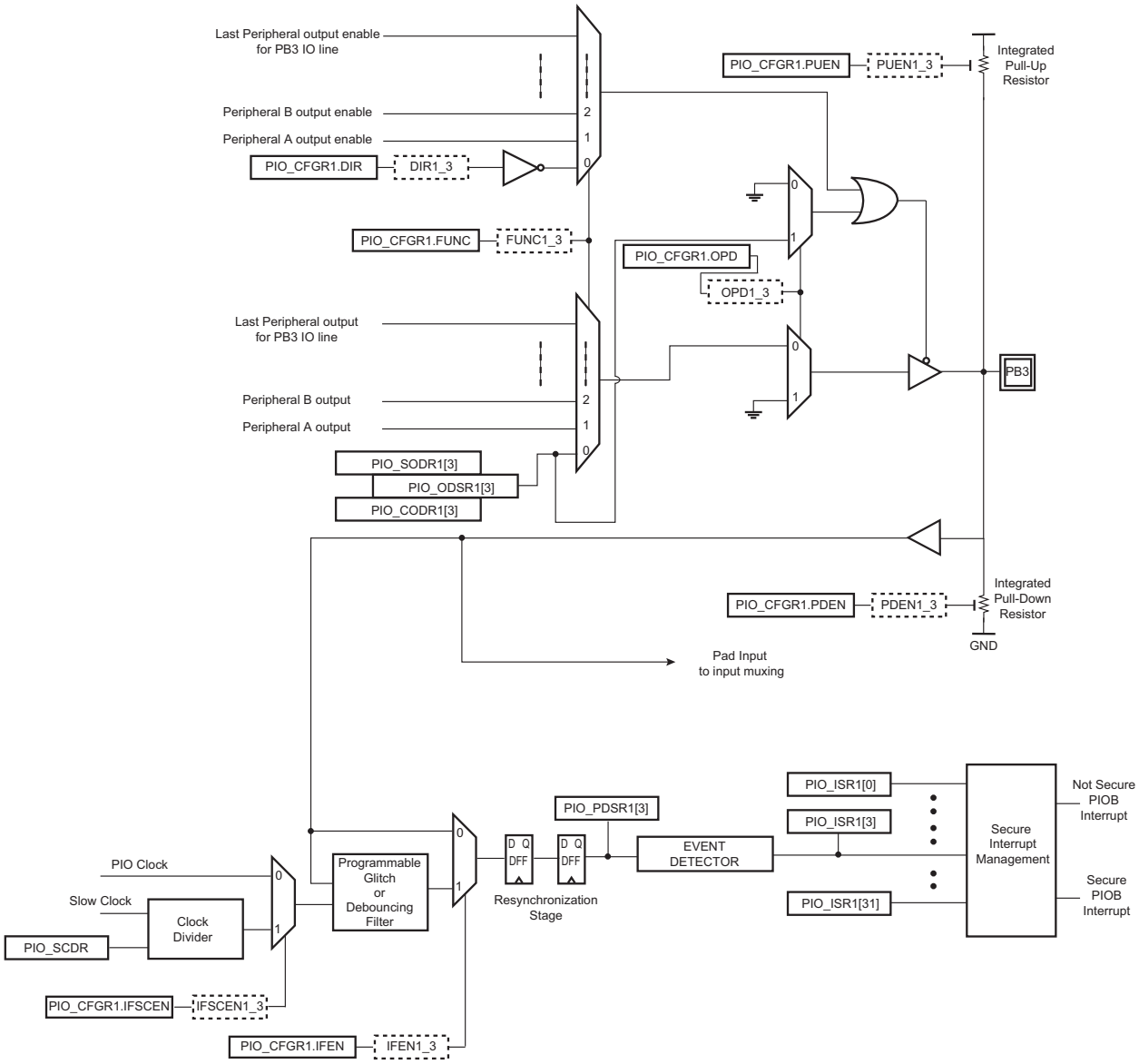
For interrupt handling, the PIO Controllers are considered as user peripherals. This means that the PIO Controller interrupt lines are connected among the interrupt sources. The PIO Controller supplies one interrupt signal per I/O group. Refer to the PIO Controller peripheral identifier in the product description to identify the interrupt sources dedicated to the PIO Controller. The PIO Controller can target either the Secure or Non-Secure Interrupt Controller according to security level of the I/O line which triggers the interruption. Using the PIO Controller requires the Interrupt Controller to be programmed first.

The PIO Controller interrupt can be generated only if the PIO Controller clock is enabled.

## 33.5 Functional Description

The PIO Controller features up to 512 fully-programmable I/O lines. Most of the control logic associated to each I/O is represented in the following figure, where the I/O line 3 of the PIOB (PB3) is described as an example. In this description each signal shown represents one of up to 512 possible indexes.

**Figure 33-2. I/O Line Control Logic**




### 33.5.1 I/O Line Configuration Method

The user interface of the PIO Controller provides several sets of registers. Each set of registers interfaces with one I/O group.

**Table 33-1. I/O Group List**

| I/O Group Number | PIO |
|---|---|
| 0 | PIOA |
| 1 | PIOB |
| ... | ... |
| 3 | PIOD |

#### 33.5.1.1 Security Management

The user must first define the security level of the I/O line. Each I/O line of each I/O group can be defined as either secure or non-secure lines. Each I/O line of the I/O group x can be set as non-secure I/O line by writing a 1 to the corresponding bit P0–P31 of the Secure PIO Set I/O Non-Secure register (S_PIO_SIONRx) of the I/O group x.

To define an I/O line of I/O group x as a secure I/O line, write a 1 to the corresponding bit P0–P31 of the Secure PIO Set I/O Secure register (S_PIO_SIOSRx) of the I/O group x.

Examples:

Setting the I/O line PC4 as non-secure line:

Write the value 16 (bit No. 4 at 1) at address 0x10B0 (S_PIO_SIONR2)

Setting the I/O line PB3 as secure line:

Write the value 8 (bit No. 3 at 1) at address 0x1074 (S_PIO_SIOSR1)

The security level of each I/O line is reported by the Secure PIO I/O Security Status register (S_PIO_IOSSRx) of the corresponding I/O group. Reading 0 at the corresponding bit P0–P31 means that the corresponding I/O line of the I/O group is defined as secure. Reading 1 means that this I/O line of the I/O group is non-secure.

The PIO Controller user interface is divided into two register mapping areas:

- The Non-Secure area, located from address 0x0 to 0x1000, can be accessed by any master (Secure or Non-Secure master). This area interfaces with all the I/O lines defined as non-secure. Trying to access to an I/O line defined as secure through this area will have no effect on I/O line and read values will be 0.
- The Secure area, located above address 0xSEC_GRP_MSB000, can only be accessed by a Secure master (if the PIO Controller is defined as secure at the HMATRIX level). This area interfaces with all the I/O lines defined as secure. Trying to access to an I/O line defined as non-secure through this area will have no effect on I/O line and read values will be 0.

#### 33.5.1.2 Programming I/O Line Configuration

The user must first define which I/O line in the group will be targeted by writing a 1 to the corresponding bit in the PIO Mask Register (PIO_MSKRx). Several I/O lines in an I/O group can be configured at the same time by setting the corresponding bits in PIO_MSKRx. Then, writing the PIO Configuration Register (PIO_CFGRx) apply the configuration to the I/O line(s) defined in PIO_MSKRx. All the I/O lines defined as secure in the S_PIO_SIOSRx must be configured by writing the S_PIO_CFGRx and S_PIO_MSKRx registers.

For more details concerning the I/O line configuration using PIO_MSKRx and PIO_CFGRx, see section I/O Lines Programming Example.

#### 33.5.1.3 Reading the I/O Line Configuration

As for programming operation, reading configuration requires the user to first define which I/O line in the group x will be targeted by writing a 1 to the corresponding bit in the PIO Mask Register (PIO_MSKRx). The value of the targeted I/O line is read in PIO_CFGRx.

If several bits are set in PIO_MSKRx, then the read configuration in PIO_CFGRx is the configuration of the I/O line with the lowest index.

Note that S_PIO_MSKRx and S_PIO_CFGRx must be used to read the configuration of a secure I/O line.

### 33.5.2 Pull-up and Pull-down Resistor Control

Each I/O line is designed with an embedded pull-up resistor and an embedded pull-down resistor.

The pull-up resistor on the I/O line(s) defined in PIO_MSKRx can be enabled by setting the PUEN bit in PIO_CFGRx. Clearing the PUEN bit in PIO_CFGRx disables the pull-up resistor of I/O lines defined in PIO_MSKRx.

The pull-down resistor on the I/O line(s) defined in PIO_MSKRx can be enabled by setting the PDEN bit in PIO_CFGRx. Clearing the PDEN bit in PIO_CFGRx disables the pull-down resistor of I/O lines defined in PIO_MSKRx.

If both PUEN and PDEN bit are set in PIO_CFGRx, only the pull-up resistor is enabled for I/O line(s) defined in PIO_MSKRx and the PDEN bit is discarded.

Control of the pull-up resistor is possible regardless of the configuration of the I/O line (Input, Output, Open-drain).

Note that S_PIO_MSKRx and S_PIO_CFGRx must be used to program the pull-up or pull-down configuration of a secure I/O line.

For more details concerning Pull-up and Pull-down configuration, see PIO Configuration Register or Secure PIO Configuration Register for secure I/O line configuration.

The reset value of PUEN and PDEN bits of each I/O line is defined at the product level and depends on the multiplexing of the device.

### 33.5.3 General Purpose or Peripheral Function Selection

The PIO Controller provides multiplexing of up to 6 peripheral functions on a single pin. The selection is performed by writing the FUNC field in PIO_CFGRx. The selected function is applied to the I/O line(s) defined in PIO_MSKRx.

When FUNC is 0, no peripheral is selected and the General Purpose PIO (GPIO) mode is selected (in this mode, the I/O line is controlled by the PIO Controller).

When FUNC is not 0, the peripheral selected to control the I/O line depends on the FUNC value.

Note that S_PIO_MSKRx and S_PIO_CFGRx must be used to program the FUNC field of a secure I/O line.

For more details, see PIO Configuration Register or Secure PIO Configuration Register for secure I/O line configuration.

Note that multiplexing of peripheral lines affects both input and output peripheral lines. When a peripheral is not selected on any I/O line, its inputs are assigned with constant default values defined at the product level. The user must ensure that only one I/O line is affected to a peripheral input at a time.

The reset value of the FUNC field of each I/O line is defined at the product level and depends on the multiplexing of the device.

### 33.5.4 Output Control

When the I/O line is assigned to a peripheral function, i.e., the corresponding FUNC field of the line configuration is not 0, the drive of the I/O line is controlled by the peripheral. According to the FUNC value, the selected peripheral determines whether the pin is driven or not.

When the FUNC field of a I/O line is 0, then the I/O line is set in General Purpose mode and the I/O line can be configured to be driven by the PIO Controller instead of the peripheral.

If the DIR bit of the I/O line configuration (PIO_CFGRx) is set (OUTPUT) then the I/O line can be driven by the PIO Controller. The level driven on an I/O line can be determined by writing in the PIO Set Output Data Register (PIO_SODRx) and the PIO Clear Output Data Register (PIO_CODRx). These write operations, respectively, set and clear the PIO Output Data Status Register (PIO_ODSRx), which represents the data driven on the I/O lines. Writing PIO_ODSRx directly is possible and only affects the I/O line set to 1 in PIO_MSKRx (see Synchronous Data Output).

When the DIR bit of the I/O line configuration is at zero, the corresponding I/O line is used as an input only.

The DIR bit has no effect if the corresponding line is assigned to a peripheral function, but writing the DIR bit is managed whether the pin is configured to be controlled by the PIO Controller or assigned to a peripheral function. This enables configuration of the I/O line prior to setting it to be managed by the PIO Controller.

Similarly, writing in PIO_SODRx and PIO_CODRx affects PIO_ODSRx. This is important as it defines the first level driven on the I/O line.

### 33.5.5 Synchronous Data Output

Clearing one or more PIO line(s) and setting another one or more PIO line(s) synchronously cannot be done by using PIO_SODRx and PIO_CODRx. It requires two successive write operations into two different registers. To overcome this, the PIO Controller offers a direct control of PIO outputs by single write access to PIO_ODSRx. Only I/O lines set to 1 in PIO_MSKRx are written.

### 33.5.6 Open-Drain Mode

Each I/O can be independently programmed in Open-Drain mode. This feature permits several drivers to be connected on the I/O line which is driven low only by each device. An external pull-up resistor (or enabling of the internal one) is generally required to guarantee a high level on the line.

The Open-Drain mode is controlled by the OPD bit in the I/O line configuration (PIO_CFGRx). An I/O line is switched in Open-Drain mode by setting the PIO_CFGRx.OPD bit. The Open-Drain mode can be selected if the I/O line is not controlled by a peripheral (the FUNC field must be cleared in PIO_CFGRx).

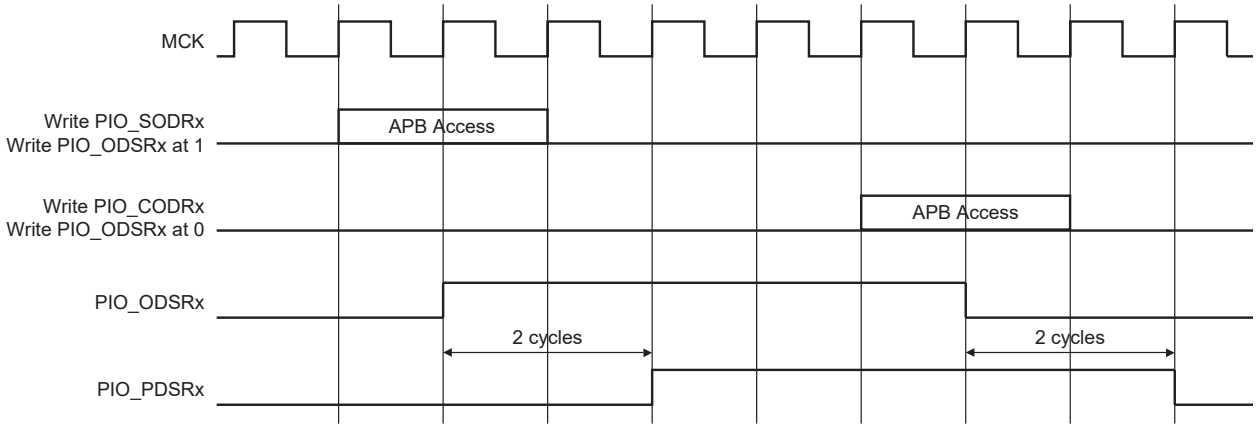
For more details concerning the Open-Drain mode, see PIO Configuration Register or Secure PIO Configuration Register for secure I/O line configuration.

After reset, the OPD bit of each I/O line is defined at the product level and depends on the multiplexing of the device.

### 33.5.7 Output Line Timings

The figure below shows how the outputs are driven either by writing PIO_SODRx or PIO_CODRx, or by directly writing PIO_ODSRx. This last case is valid only if the corresponding bit in PIO_MSKRx is set. The figure also shows when the feedback in the Pin Data Status Register (PIO_PDSRx) is available.

**Figure 33-3. Output Line Timings**




### 33.5.8 Inputs

The level on each I/O line of the I/O group x can be read through PIO_PDSRx. This register indicates the level of the I/O lines regardless of their configuration, whether uniquely as an input, or driven by the PIO Controller, or driven by a peripheral.

Reading the I/O line levels requires the clock of the PIO Controller to be enabled, otherwise PIO_PDSRx reads the levels present on the I/O line at the time the clock was disabled.

### 33.5.9 Input Glitch and Debouncing Filters

Optional input glitch and debouncing filters are independently programmable on each I/O line.

The glitch filter can filter a glitch with a duration of less than 1/2 master clock (MCK) and the debouncing filter can filter a pulse of less than 1/2 period of a programmable divided slow clock.

The selection between glitch filtering or debounce filtering is done by writing the bit IFSCEN in the PIO Configuration Register (PIO_CFGRx). The selected filtering mode is applied to the I/O line(s) defined in PIO_MSKRx.

- If IFSCEN = 0: The glitch filter can filter a glitch with a duration of less than 1/2 master clock period.
- If IFSCEN = 1: The debouncing filter can filter a pulse with a duration of less than 1/2 programmable divided slow clock period.

For the debouncing filter, the period of the divided slow clock is performed by writing in the DIV field of the Secure PIO Slow Clock Divider Debouncing Register (S_PIO_SCDR): $t_{div_slck} = ((DIV + 1) \times 2) \times t_{slck}$.

When the glitch or debouncing filter is enabled, a glitch or pulse with a duration of less than 1/2 selected clock cycle (selected clock represents MCK or divided slow clock depending on IFSCEN bit programming) is automatically rejected, while a pulse with a duration of one selected clock (MCK or divided slow clock) cycle or more is accepted. For pulse durations between 1/2 selected clock cycle and one selected clock cycle, the pulse may or may not be taken into account, depending on the precise timing of its occurrence. Thus for a pulse to be visible, it must exceed one selected clock cycle, whereas for a glitch to be reliably filtered out, its duration must not exceed 1/2 selected clock cycle.

The filters also introduce some latencies, illustrated in the figures below .

The glitch filter of each I/O lines is controlled by the IFEN bit of the PIO Configuration Register (PIO_CFGRx). Setting the PIO_CFGRx.IFEN bit enables the glitch filter of the I/O line(s) defined in PIO_MSKRx.

When the glitch and/or debouncing filter is enabled, it does not modify the behavior of the inputs on the peripherals. It acts only on the value read in PIO_PDSRx and on the input change interrupt detection. The glitch and debouncing filters require that the PIO Controller clock is enabled.

**Figure 33-4. Input Glitch Filter Timing**

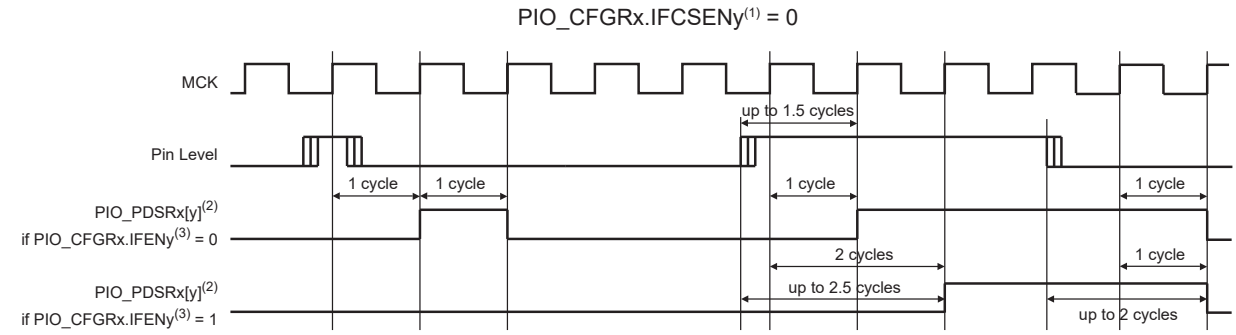
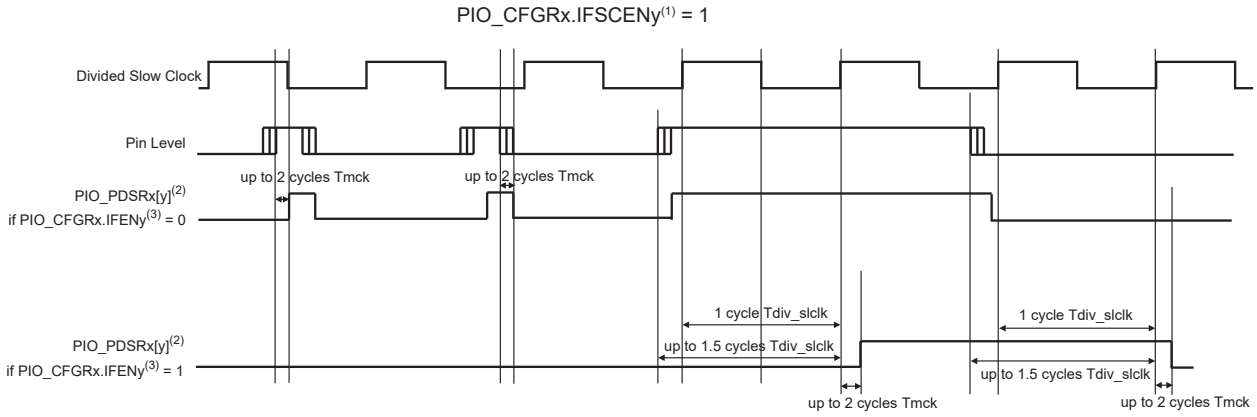



**Figure 33-5. Input Debouncing Filter Timing**




**Note:**

1. Means IFCSEN bit of the I/O line y of the I/O group x.
2. Means PIO Data Status value of the I/O line y of the I/O group x.
3. Means IFEN bit of the I/O line y of the I/O group x.

### 33.5.10 Input Edge/Level Interrupt

Each I/O group can be programmed to generate an interrupt when it detects an edge or a level on an I/O line. The Input Edge/Level interrupts are controlled by writing the PIO Interrupt Enable Register (PIO_IERx) and the PIO Interrupt Disable Register (PIO_IDRx), which enable and disable the input change interrupt respectively by setting and clearing the corresponding bit in the PIO Interrupt Mask Register (PIO_IMRx). For the Secure I/O lines, the Input Edge/Level interrupts are controlled by writing S_PIO_IERx and S_PIO_IDRx, which enable and disable input change interrupts respectively by setting and clearing the corresponding bit in the S_PIO_IMRx. As input change detection is possible only by comparing two successive samplings of the input of the I/O line, the PIO Controller clock must be enabled. The Input Change interrupt is available regardless of the configuration of the I/O line, i.e., configured as an input only, controlled by the PIO Controller or assigned to a peripheral function.

Each I/O group can generate a Non-Secure interrupt and a Secure interrupt according to the security level of the I/O line which triggers the interrupt.

According to the EVTSEL field value in the PIO Configuration Register (PIO_CFGRx) or the Secure PIO Configuration Register (S_PIO_CFGRx) in case of a Secure I/O line, the interrupt signal of the I/O group x can be generated on the following occurrence:

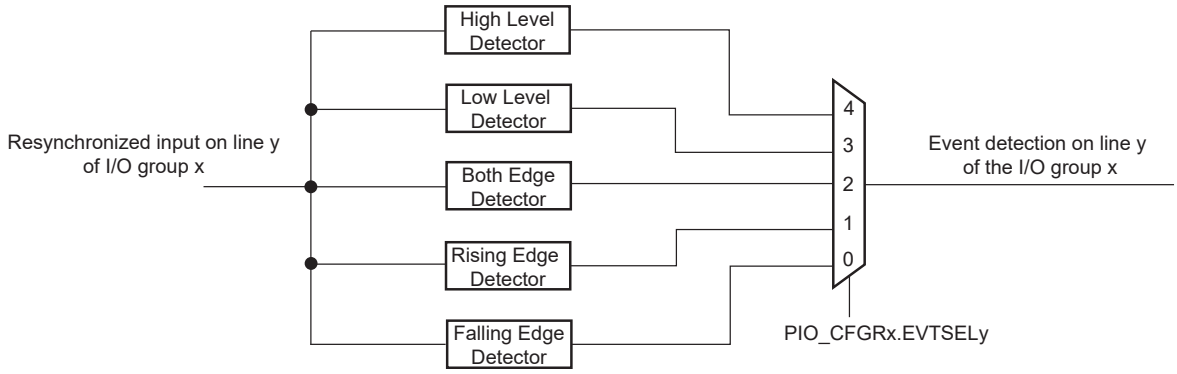
- (S_)PIO_CFGRx.EVTSELy = 0: The interrupt signal of the I/O group x is generated on the I/O line y falling edge detection (assuming that (S_)PIO_IMRx[y] = 1).
- (S_)PIO_CFGRx.EVTSELy = 1: The interrupt signal of the I/O group x is generated on the I/O line y rising edge detection (assuming that (S_)PIO_IMRx[y] = 1).
- (S_)PIO_CFGRx.EVTSELy = 2: The interrupt signal of the I/O group x is generated on the I/O line y both rising and falling edge detection (assuming that (S_)PIO_IMRx[y] = 1).
- (S_)PIO_CFGRx.EVTSELy = 3: The interrupt signal of the I/O group x is generated on the I/O line y low level detection (assuming that (S_)PIO_IMRx[y] = 1).
- (S_)PIO_CFGRx.EVTSELy = 4: The interrupt signal of the I/O group x is generated on the I/O line y high level detection (assuming that (S_)PIO_IMRx[y] = 1).

By default, the interrupt can be generated at any time a falling edge is detected on the input.

When an input edge or level is detected on an I/O line, the corresponding bit in the PIO Interrupt Status Register (PIO_ISRx), or in the Secure PIO Interrupt Status Register (S_PIO_ISRx) if the I/O line is Secure, is set. For a Non-Secure I/O line, if the corresponding bit in PIO_IMRx is set, the PIO Controller Non-Secure interrupt line of the I/O group x is asserted. For a Secure I/O line, if the corresponding bit in S_PIO_IMRx is set, the PIO Controller Secure interrupt line of the I/O group x is asserted.

When the software reads PIO_ISRx, all the Non-Secure interrupts of the I/O group x are automatically cleared. When the software reads S_PIO_ISRx, all the Secure interrupts of the I/O group x are automatically cleared. This signifies that all the interrupts that are pending when PIO_ISRx or S_PIO_ISRx is read must be handled. When an Interrupt is enabled on a “level”, the interrupt is generated as long as the interrupt source is not cleared, even if some read accesses in PIO_ISRx or S_PIO_ISRx are performed.

**Figure 33-6. Event Detector on Input Lines**




Example of interrupt generation on following lines:

- Rising edge on the Secure PIO line 0 of the I/O group 0 (PIOA)
- Low-level edge on the Secure PIO line 1of the I/O group 0 (PIOA)
- Rising edge on the Secure PIO line 2 of the I/O group 0 (PIOA)
- High-level on the Secure PIO line 3 of the I/O group 0 (PIOA)
- Low-level on the Non-Secure PIO line 4 of the I/O group 0 (PIOA)
- High-level on the Secure PIO line 0 of the I/O group 1 (PIOB)
- Falling edge on the Secure PIO line 1 of the I/O group 1 (PIOB)

© 2018 Microchip Technology Inc.

- Rising edge on the Secure PIO line 2 of the I/O group 1 (PIOB)
- Any edge on the other Non-Secure lines of the I/O group 1 (PIOB)

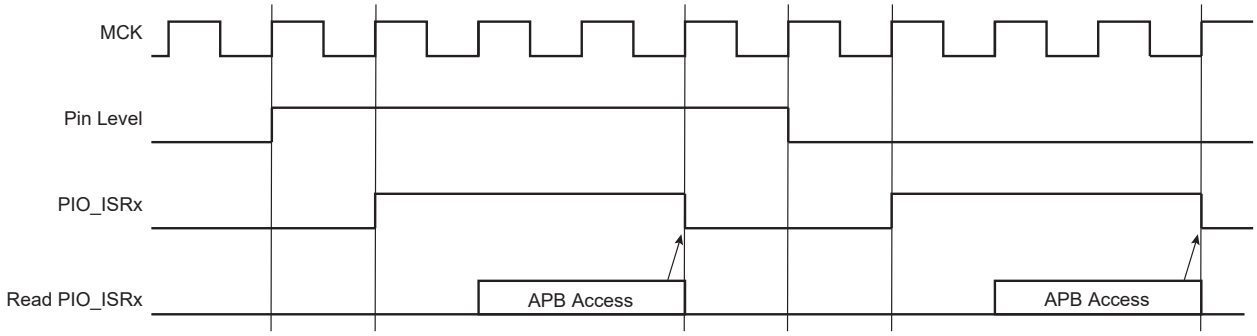
The table below details the required configuration for this example.

**Table 33-2. Configuration for Example Interrupt Generation**

| Configuration | Name |
|---|---|
| PIOA: I/O line security level | Define the I/O lines 0 to 3 of the PIOA as Secure by writing 32'h0000_000F in the S_PIO_SIOSR0 (offset 0x1034)<br>Define the I/O lines 4 of the PIOA as Non-Secure by writing 32'h0000_0010 in the S_PIO_SIONR0 (offset 0x1030) |
| PIOA: Interrupt Mode | Enable interrupt sources for lines 0 to 3 of PIOA by writing 32'h0000_000F in S_PIO_IER0 (offset 0x1020)<br>Enable interrupt source for the line 4 of PIOA by writing 32'h0000_0010 in PIO_IER0 (offset 0x20) |
| PIOA: Event Selection | Configure Rising Edge detection for Secure lines 0 and 2:<br>Write 32'h0000_0005 in S_PIO_MSKR0 (offset 0x1000)<br>Write 32'h0100_0000 in S_PIO_CFGR0 (offset 0x1004) |
| | Configure Low Level detection for Secure line 1:<br>Write 32'h0000_0002 in S_PIO_MSKR0 (offset 0x1000)<br>Write 32'h0300_0000 in S_PIO_CFGR0 (offset 0x1004) |
| | Configure High Level detection for Secure line 3:<br>Write 32'h0000_0008 in S_PIO_MSKR0 (offset 0x1000)<br>Write 32'h0400_0000 in S_PIO_CFGR0 (offset 0x1004) |
| | Configure Low Level detection for Non-Secure line 4:<br>Write 32'h0000_0010 in PIO_MSKR0 (offset 0x0)<br>Write 32'h0300_0000 in PIO_CFGR0 (offset 0x4) |
| PIOB: I/O line security level | Define the I/O lines 0 to 2 of the PIOB as Secure by writing 32'h0000_0007 in the S_PIO_SIOSR1 (offset 0x1074)<br>Define the other I/O lines of the PIOB as Non-Secure by writing 32'hFFFF_FFF8 in the S_PIO_SIONR1 (offset 0x1070) |
| PIOB: Interrupt Mode | Enable interrupt sources for lines 0 to 2 of PIOB by writing 32'h0000_0007 in S_PIO_IER1 (offset 0x1060)<br>Enable interrupt sources for all other lines of PIOB by writing 32'hFFFF_FFF8 in PIO_IER1 (offset 0x60) |

| .....continued | |
|---|---|
| **Configuration** | **Name** |
| PIOB: Event Selection | Configure High Level detection for Secure line 0:<br>Write 32'h0000_0001 in S_PIO_MSKR1 (offset 0x1040)<br>Write 32'h0400_0000 in S_PIO_CFGR1 (offset 0x1044) |
| | Configure Falling Edge detection for Secure line 1:<br>Write 32'h0000_0002 in S_PIO_MSKR1 (offset 0x1040)<br>Write 32'h0000_0000 in S_PIO_CFGR1 (offset 0x1044) |
| | Configure Rising Edge detection for Secure line 2:<br>Write 32'h0000_0004 in S_PIO_MSKR1 (offset 0x1040)<br>Write 32'h0100_000 in S_PIO_CFGR1 (offset 0x1044) |
| | Configure Low Level detection for Non-Secure lines:<br>Write 32'hFFFF_FFF8 in PIO_MSKR1 (offset 0x40)<br>Write 32'h0200_000 in PIO_CFGR1 (offset 0x44) |

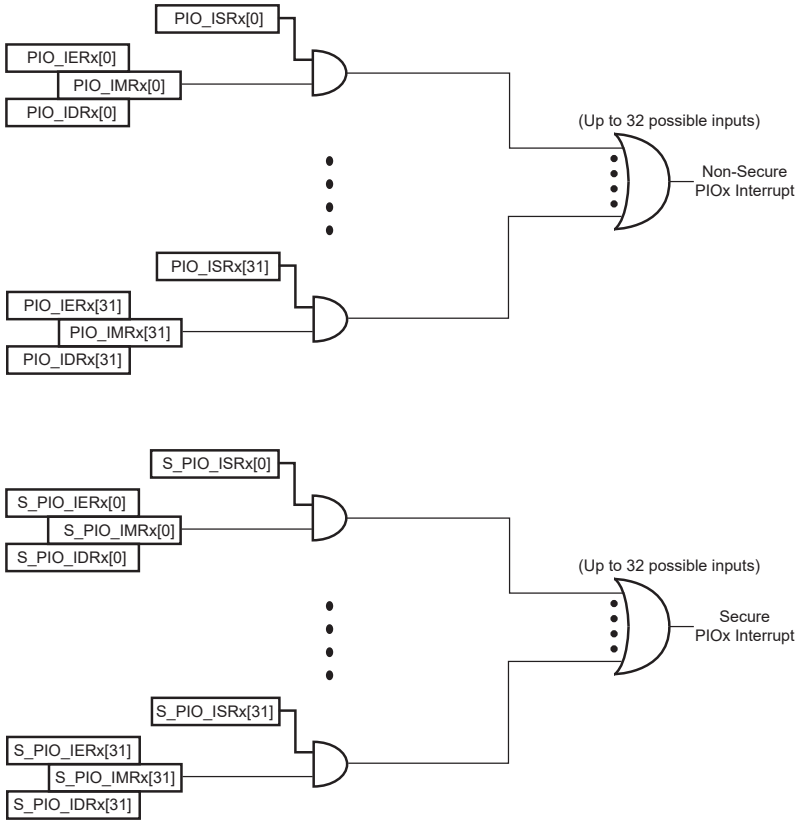
**Figure 33-7. Input Change Interrupt Timings When No Additional Interrupt Modes**




### 33.5.11 Interrupt Management

The PIO Controller can drive one secure interrupt signal and one non-secure interrupt signal per I/O group (refer to the block diagram). Secure interrupt signals are connected to the secure interrupt controller of the system. Non-secure interrupt signals are connected to the non-secure interrupt controller of the system.

**Figure 33-8. PIO Interrupt Management**




### 33.5.12 I/O Lines Lock

When an I/O line is controlled by a peripheral (particularly the Pulse Width Modulation Controller PWM), it can become locked by the action of this peripheral via an input of the PIO Controller. When an I/O line is locked, the following fields in PIO_CFGRx are locked and cannot be modified:

- FUNC: Peripheral selection cannot be changed when the corresponding I/O line is locked.
- PUEN: Pull-Up configuration cannot be changed when the corresponding I/O line is locked.
- PDEN: Pull-Down configuration cannot be changed when the corresponding I/O line is locked.
- OPD: Open Drain configuration cannot be changed when the corresponding I/O line is locked.

Writing to one of these fields while the corresponding I/O line is locked will have no effect.

The user can know at anytime which I/O line is locked by reading the PIO Lock Status Register (PIO_LOCKSR) or Secure PIO Lock Status Register (S_PIO_LOCKSR) for locked Secure I/O lines. Once an I/O line is locked, the only way to unlock it is to apply a hardware reset to the PIO Controller.

### 33.5.13 Programmable I/O Drive

It is possible to configure the I/O drive for pads PA0 to PD31. The I/O drive of the pad can be programmed by writing the DRVSTR field in the PIO Configuration Register (PIO_CFGRx) if the corresponding line is Non-Secure or the Secure PIO Configuration Register (S_PIO_CFGRx) if the I/O line is Secure. For any details, refer to the product electrical characteristics.

### 33.5.14 Programmable Schmitt Trigger

It is possible to configure each input for the Schmitt trigger. The Schmitt trigger can be enabled by setting the SCHMITT bit of the PIO Configuration Register (PIO_CFGRx) if the corresponding line is Non-Secure

or the Secure PIO Configuration Register (S_PIO_CFGRx) if the I/O line is Secure. By default the Schmitt trigger is active. Disabling the Schmitt trigger is requested when using the QTouch Library.

### 33.5.15 I/O Line Configuration Freeze

#### 33.5.15.1 Software Freeze

Once the I/O line configuration is done, it can be frozen by using the PIO I/O Freeze Configuration Register (PIO_IOFRx) of the corresponding group or the Secure PIO I/O Freeze Configuration Register (S_PIO_IOFRx) if the I/O line is Secure.

##### 33.5.15.1.1 Physical Freeze

Setting the FPHY bit of PIO_IOFRx freezes the following fields of the Non-Secure I/O lines defined in PIO_MSKRx:

- FUNC: I/O Line Function
- DIR: Direction
- PUEN: Pull-Up Enable
- PDEN: Pull-Down Enable
- OPD: Open-Drain
- SCHMITT: Schmitt Trigger
- DRVSTR: Drive Strength

For Secure I/O lines, use the FPHY bit of the S_PIO_IOFRx and the S_PIO_MSKRx to freeze the fields above.

When the physical freeze is currently active on an I/O line, the PCFS flag is set when reading the PIO_CFGRx of the I/O line (or the S_PIO_CFGRx if the I/O line is Secure).

Only a hardware reset can release fields listed above.

##### 33.5.15.1.2 Interrupt Freeze

Setting the FINT bit of the PIO_IOFRx will freeze the following fields of the Non-Secure I/O lines defined in the PIO_MSKRx:

- IFEN: Input Filter Enable
- IFSCEN: Input Filter Slow Clock Enable
- EVTSEL: Event Selection

For Secure I/O lines, use the FINT bit of the S_PIO_IOFRx and the S_PIO_MSKRx to freeze the fields above.

When the “Interrupt Freeze” is currently active on an I/O line, the ICFS flag is set when reading the PIO_CFGRx of the I/O line (or the S_PIO_CFGRx if the I/O line is Secure).

Only a hardware reset can release fields listed above.

### 33.5.16 Register Write Protection

To prevent any single software error from corrupting PIO behavior, certain registers in the address space can be write-protected by setting WPEN in the PIO Write Protection Mode Register (PIO_WPMR) or the Secure PIO Write Protection Mode Register (S_PIO_WPMR).

If a write access to a Non-Secure write-protected register is detected, the WPVS flag in the PIO Write Protection Status Register (PIO_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

If a write access to a Secure write-protected register is detected, the WPVS flag in the Secure PIO Write Protection Status Register (S_PIO_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The respective WPVS bit is automatically cleared after reading the PIO_WPSR or S_PIO_WPSR.

The following registers are write-protected when WPEN is set in PIO_WPMR:

- PIO Mask Register
- PIO Configuration Register

The following registers are write-protected when WPEN is set in S_PIO_WPMR:

- Secure PIO Mask Register
- Secure PIO Configuration Register
- Secure PIO Slow Clock Divider Debouncing Register

The following registers are write-protected when WPITEN is set in S_PIO_WPMR:

- Secure PIO Interrupt Enable Register
- Secure PIO Interrupt Disable Register

## 33.6 I/O Lines Programming Example

The programming example shown in the table below is used to obtain the following configurations:

- PIOA Configuration:
  - 4-bit output port on Secure I/O lines 0 to 3, open-drain, with pull-up resistor
  - Four output signals on Non-Secure I/O lines 4 to 7 (to drive LEDs for example), driven high and low, no pull-up resistor, no pull-down resistor
  - Secure I/O lines 16 to 19 assigned to peripheral A functions with pull-up resistor
  - Non-Secure I/O lines 20 to 23 assigned to peripheral B functions with pull-down resistor
- PIOB Configuration:
  - Four input signals on Secure I/O lines 0 to 3 (to read push-button states for example), with pull-up resistors, glitch filters and input change interrupts
  - Four input signals on Non-Secure I/O lines 12 to 15 to read an external device status (polled, thus no input change interrupt), no pull-up resistor, no glitch filter
  - Secure I/O lines 16 to 23 assigned to peripheral B functions with pull-down resistor
  - Non-Secure I/O lines 24 to 27 assigned to peripheral D with Input Change Interrupt, no pull-up resistor and no pull-down resistor

**Table 33-3. Programming Example**

| Action | Register | Value to be Written |
|---|---|---|
| PIOA: Set I/O lines 0 to 3 and 16 to 19 as Secure | S_PIO_SIOSR0 (offset 0x1034) | 0x000F_000F |
| PIOA: Set I/O lines 4 to 7 and 20 to 23 as Non-Secure | S_PIO_SIONR0 (offset 0x1030) | 0x00F0_00F0 |

.....continued

| Action | Register | Value to be Written |
|---|---|---|
| PIOA: 4-bit output port on Secure I/O lines 0 to 3, open-drain, with pull-up resistor | S_PIO_MSKR0 (offset 0x1000) | 0x0000_000F |
| | S_PIO_CFGR0 (offset 0x1004) | 0x0000_4300 |
| PIOA: Four output signals on Non-Secure I/O lines 4 to 7 (to drive LEDs for example), driven high and low, no pull-up resistor, no pull-down resistor | PIO_MSKR0 (offset 0x0) | 0x0000_00F0 |
| | PIO_CFGR0 (offset 0x4) | 0x0000_0100 |
| PIOA: Secure I/O lines 16 to 19 assigned to peripheral A functions with pull-up resistor | S_PIO_MSKR0 (offset 0x1000) | 0x000F_0000 |
| | S_PIO_CFGR0 (offset 0x1004) | 0x0000_0201 |
| PIOA: Non-Secure I/O lines 20 to 23 assigned to peripheral B functions with pull-down resistor | PIO_MSKR0 (offset 0x0) | 0x00F0_0000 |
| | PIO_CFGR0 (offset 0x4) | 0x0000_0402 |
| PIOB: Set I/O lines 0 to 3 and 16 to 23as Secure | S_PIO_SIOSR1 (offset 0x1074) | 0x00FF_000F |
| PIOB: Set I/O lines 12 to 15 and 24 to 27 as Non-Secure | S_PIO_SIONR1 (offset 0x1070) | 0x0F00_F000 |
| PIOB: Four input signals on Secure I/O lines 0 to 3 (to read push-button states for example), with pull-up resistors, glitch filters and interrupts on rising edge | S_PIO_MSKR1 (offset 0x1040) | 0x0000_000F |
| | S_PIO_CFGR1 (offset 0x1044) | 0x0100_1200 |
| PIOB: Four input signals on Non-Secure I/O line 12 to 15 to read an external device status (polled, thus no input change interrupt), no pull-up resistor, no glitch filter | PIO_MSKR1 (offset 0x40) | 0x0000_F000 |
| | PIO_CFGR1 (offset 0x44) | 0x0100_1200 |
| PIOB: Secure I/O lines 16 to 23 assigned to peripheral B functions with pull-down resistor | S_PIO_MSKR1 (offset 0x1040) | 0x00FF_0000 |
| | S_PIO_CFGR1 (offset 0x1044) | 0x0000_0402 |

| .....continued | | |
|---|---|---|
| **Action** | **Register** | **Value to be Written** |
| PIOB: Non-Secure I/O line 24 to 27 assigned to peripheral D with Input Interrupt on both edges, no pull-up resistor and no pull-down resistor | PIO_MSKR1 (offset 0x40) | 0x0F00_0000 |
| | PIO_CFGR1 (offset 0x44) | 0x0200_0004 |
| PIOB: Enable interrupt | S_PIO_IER1 (offset 0x1060) | 0x0000_000F |
| | PIO_IER1 (offset 0x60) | 0x0F00_0000 |

## 33.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | PIO_MSKR0 | 7:0 | MSK7 | MSK6 | MSK5 | MSK4 | MSK3 | MSK2 | MSK1 | MSK0 |
| | | 15:8 | MSK15 | MSK14 | MSK13 | MSK12 | MSK11 | MSK10 | MSK9 | MSK8 |
| | | 23:16 | MSK23 | MSK22 | MSK21 | MSK20 | MSK19 | MSK18 | MSK17 | MSK16 |
| | | 31:24 | MSK31 | MSK30 | MSK29 | MSK28 | MSK27 | MSK26 | MSK25 | MSK24 |
| 0x04 | PIO_CFGR0 | 7:0 | | | | | | FUNC[2:0] | | |
| | | 15:8 | SCHMITT | OPD | IFSCEN | IFEN | | PDEN | PUEN | DIR |
| | | 23:16 | | | | | | | DRVSTR[1:0] | |
| | | 31:24 | | ICFS | PCFS | | | EVTSEL[2:0] | | |
| 0x08 | PIO_PDSR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x0C | PIO_LOCKSR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10 | PIO_SODR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x14 | PIO_CODR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x18 | PIO_ODSR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1C ... 0x1F | Reserved | | | | | | | | | |
| 0x20 | PIO_IER0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x24 | PIO_IDR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x28 | PIO_IMR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x2C ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | PIO_ISR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x34 ... 0x3B | Reserved | | | | | | | | | |
| 0x3C | PIO_IOFR0 | 7:0 | | | | | | | FINT | FPHY |
| | | 15:8 | FRZKEY[7:0] | | | | | | | |
| | | 23:16 | FRZKEY[15:8] | | | | | | | |
| | | 31:24 | FRZKEY[23:16] | | | | | | | |
| 0x40 | PIO_MSKR1 | 7:0 | MSK7 | MSK6 | MSK5 | MSK4 | MSK3 | MSK2 | MSK1 | MSK0 |
| | | 15:8 | MSK15 | MSK14 | MSK13 | MSK12 | MSK11 | MSK10 | MSK9 | MSK8 |
| | | 23:16 | MSK23 | MSK22 | MSK21 | MSK20 | MSK19 | MSK18 | MSK17 | MSK16 |
| | | 31:24 | MSK31 | MSK30 | MSK29 | MSK28 | MSK27 | MSK26 | MSK25 | MSK24 |
| 0x44 | PIO_CFGR1 | 7:0 | | | | | | FUNC[2:0] | | |
| | | 15:8 | SCHMITT | OPD | IFSCEN | IFEN | | PDEN | PUEN | DIR |
| | | 23:16 | | | | | | | DRVSTR[1:0] | |
| | | 31:24 | | ICFS | PCFS | | | EVTSEL[2:0] | | |
| 0x48 | PIO_PDSR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x4C | PIO_LOCKSR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x50 | PIO_SODR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x54 | PIO_CODR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x58 | PIO_ODSR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x5C ... 0x5F | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x60 | PIO_IER1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x64 | PIO_IDR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x68 | PIO_IMR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x6C ... 0x6F | Reserved | | | | | | | | | |
| 0x70 | PIO_ISR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x74 ... 0x7B | Reserved | | | | | | | | | |
| 0x7C | PIO_IOFR1 | 7:0 | | | | | | | FINT | FPHY |
| | | 15:8 | FRZKEY[7:0] | | | | | | | |
| | | 23:16 | FRZKEY[15:8] | | | | | | | |
| | | 31:24 | FRZKEY[23:16] | | | | | | | |
| 0x80 | PIO_MSKR2 | 7:0 | MSK7 | MSK6 | MSK5 | MSK4 | MSK3 | MSK2 | MSK1 | MSK0 |
| | | 15:8 | MSK15 | MSK14 | MSK13 | MSK12 | MSK11 | MSK10 | MSK9 | MSK8 |
| | | 23:16 | MSK23 | MSK22 | MSK21 | MSK20 | MSK19 | MSK18 | MSK17 | MSK16 |
| | | 31:24 | MSK31 | MSK30 | MSK29 | MSK28 | MSK27 | MSK26 | MSK25 | MSK24 |
| 0x84 | PIO_CFGR2 | 7:0 | | | | | | FUNC[2:0] | | |
| | | 15:8 | SCHMITT | OPD | IFSCEN | IFEN | | PDEN | PUEN | DIR |
| | | 23:16 | | | | | | | DRVSTR[1:0] | |
| | | 31:24 | | ICFS | PCFS | | | EVTSEL[2:0] | | |
| 0x88 | PIO_PDSR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x8C | PIO_LOCKSR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x90 | PIO_SODR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x94 | PIO_CODR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x98 | PIO_ODSR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x9C ... 0x9F | Reserved | | | | | | | | | |
| 0xA0 | PIO_IER2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xA4 | PIO_IDR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xA8 | PIO_IMR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xAC ... 0xAF | Reserved | | | | | | | | | |
| 0xB0 | PIO_ISR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xB4 ... 0xBB | Reserved | | | | | | | | | |
| 0xBC | PIO_IOFR2 | 7:0 | | | | | | | FINT | FPHY |
| | | 15:8 | FRZKEY[7:0] | | | | | | | |
| | | 23:16 | FRZKEY[15:8] | | | | | | | |
| | | 31:24 | FRZKEY[23:16] | | | | | | | |
| 0xC0 | PIO_MSKR3 | 7:0 | MSK7 | MSK6 | MSK5 | MSK4 | MSK3 | MSK2 | MSK1 | MSK0 |
| | | 15:8 | MSK15 | MSK14 | MSK13 | MSK12 | MSK11 | MSK10 | MSK9 | MSK8 |
| | | 23:16 | MSK23 | MSK22 | MSK21 | MSK20 | MSK19 | MSK18 | MSK17 | MSK16 |
| | | 31:24 | MSK31 | MSK30 | MSK29 | MSK28 | MSK27 | MSK26 | MSK25 | MSK24 |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xC4 | PIO_CFGR3 | 7:0 | | | | | | FUNC[2:0] | | |
| | | 15:8 | SCHMITT | OPD | IFSCEN | IFEN | | PDEN | PUEN | DIR |
| | | 23:16 | | | | | | | DRVSTR[1:0] | |
| | | 31:24 | | ICFS | PCFS | | | EVTSEL[2:0] | | |
| 0xC8 | PIO_PDSR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xCC | PIO_LOCKSR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xD0 | PIO_SODR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xD4 | PIO_CODR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xD8 | PIO_ODSR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xDC ... 0xDF | Reserved | | | | | | | | | |
| 0xE0 | PIO_IER3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xE4 | PIO_IDR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xE8 | PIO_IMR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xEC ... 0xEF | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xF0 | PIO_ISR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xF4 ... 0xFB | Reserved | | | | | | | | | |
| 0xFC | PIO_IOFR3 | 7:0 | | | | | | | FINT | FPHY |
| | | 15:8 | FRZKEY[7:0] | | | | | | | |
| | | 23:16 | FRZKEY[15:8] | | | | | | | |
| | | 31:24 | FRZKEY[23:16] | | | | | | | |
| 0x0100 ... 0x05DF | Reserved | | | | | | | | | |
| 0x05E0 | PIO_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0x05E4 | PIO_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05E8 ... 0x0FFF | Reserved | | | | | | | | | |
| 0x1000 | S_PIO_MSKR0 | 7:0 | MSK7 | MSK6 | MSK5 | MSK4 | MSK3 | MSK2 | MSK1 | MSK0 |
| | | 15:8 | MSK15 | MSK14 | MSK13 | MSK12 | MSK11 | MSK10 | MSK9 | MSK8 |
| | | 23:16 | MSK23 | MSK22 | MSK21 | MSK20 | MSK19 | MSK18 | MSK17 | MSK16 |
| | | 31:24 | MSK31 | MSK30 | MSK29 | MSK28 | MSK27 | MSK26 | MSK25 | MSK24 |
| 0x1004 | S_PIO_CFGR0 | 7:0 | | | | | | FUNC[2:0] | | |
| | | 15:8 | SCHMITT | OPD | IFSCEN | IFEN | | PDEN | PUEN | DIR |
| | | 23:16 | | | | | | | DRVSTR[1:0] | |
| | | 31:24 | | ICFS | PCFS | | | EVTSEL[2:0] | | |
| 0x1008 | S_PIO_PDSR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x100C | S_PIO_LOCKSR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1010 | S_PIO_SODR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x1014 | S_PIO_CODR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1018 | S_PIO_ODSR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x101C ... 0x101F | Reserved | | | | | | | | | |
| 0x1020 | S_PIO_IER0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1024 | S_PIO_IDR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1028 | S_PIO_IMR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x102C | S_PIO_ISR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1030 | S_PIO_SIONR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1034 | S_PIO_SIOSR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1038 | S_PIO_IOSSR0 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x103C | S_PIO_IOFR0 | 7:0 | FPH7 | FPH6 | FPH5 | FPH4 | FPH3 | FPH2 | FPH1 | FPH0 |
| | | 15:8 | FPH15 | FPH14 | FPH13 | FPH12 | FPH11 | FPH10 | FPH9 | FPH8 |
| | | 23:16 | FPH23 | FPH22 | FPH21 | FPH20 | FPH19 | FPH18 | FPH17 | FPH16 |
| | | 31:24 | FPH31 | FPH30 | FPH29 | FPH28 | FPH27 | FPH26 | FPH25 | FPH24 |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x1040 | S_PIO_MSKR1 | 7:0 | MSK7 | MSK6 | MSK5 | MSK4 | MSK3 | MSK2 | MSK1 | MSK0 |
| | | 15:8 | MSK15 | MSK14 | MSK13 | MSK12 | MSK11 | MSK10 | MSK9 | MSK8 |
| | | 23:16 | MSK23 | MSK22 | MSK21 | MSK20 | MSK19 | MSK18 | MSK17 | MSK16 |
| | | 31:24 | MSK31 | MSK30 | MSK29 | MSK28 | MSK27 | MSK26 | MSK25 | MSK24 |
| 0x1044 | S_PIO_CFGR1 | 7:0 | | | | | | FUNC[2:0] | | |
| | | 15:8 | SCHMITT | OPD | IFSCEN | IFEN | | PDEN | PUEN | DIR |
| | | 23:16 | | | | | | | DRVSTR[1:0] | |
| | | 31:24 | | ICFS | PCFS | | | EVTSEL[2:0] | | |
| 0x1048 | S_PIO_PDSR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x104C | S_PIO_LOCKSR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1050 | S_PIO_SODR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1054 | S_PIO_CODR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1058 | S_PIO_ODSR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x105C ... 0x105F | Reserved | | | | | | | | | |
| 0x1060 | S_PIO_IER1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1064 | S_PIO_IDR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1068 | S_PIO_IMR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x106C | S_PIO_ISR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1070 | S_PIO_SIONR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1074 | S_PIO_SIOSR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1078 | S_PIO_IOSSR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x107C | S_PIO_IOFR1 | 7:0 | FPH7 | FPH6 | FPH5 | FPH4 | FPH3 | FPH2 | FPH1 | FPH0 |
| | | 15:8 | FPH15 | FPH14 | FPH13 | FPH12 | FPH11 | FPH10 | FPH9 | FPH8 |
| | | 23:16 | FPH23 | FPH22 | FPH21 | FPH20 | FPH19 | FPH18 | FPH17 | FPH16 |
| | | 31:24 | FPH31 | FPH30 | FPH29 | FPH28 | FPH27 | FPH26 | FPH25 | FPH24 |
| 0x1080 | S_PIO_MSKR2 | 7:0 | MSK7 | MSK6 | MSK5 | MSK4 | MSK3 | MSK2 | MSK1 | MSK0 |
| | | 15:8 | MSK15 | MSK14 | MSK13 | MSK12 | MSK11 | MSK10 | MSK9 | MSK8 |
| | | 23:16 | MSK23 | MSK22 | MSK21 | MSK20 | MSK19 | MSK18 | MSK17 | MSK16 |
| | | 31:24 | MSK31 | MSK30 | MSK29 | MSK28 | MSK27 | MSK26 | MSK25 | MSK24 |
| 0x1084 | S_PIO_CFGR2 | 7:0 | | | | | | FUNC[2:0] | | |
| | | 15:8 | SCHMITT | OPD | IFSCEN | IFEN | | PDEN | PUEN | DIR |
| | | 23:16 | | | | | | | DRVSTR[1:0] | |
| | | 31:24 | | ICFS | PCFS | | | EVTSEL[2:0] | | |
| 0x1088 | S_PIO_PDSR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x108C | S_PIO_LOCKSR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1090 | S_PIO_SODR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1094 | S_PIO_CODR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x1098 | S_PIO_ODSR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x109C ... 0x109F | Reserved | | | | | | | | | |
| 0x10A0 | S_PIO_IER2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10A4 | S_PIO_IDR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10A8 | S_PIO_IMR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10AC | S_PIO_ISR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10B0 | S_PIO_SIONR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10B4 | S_PIO_SIOSR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10B8 | S_PIO_IOSSR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10BC | S_PIO_IOFR2 | 7:0 | FPH7 | FPH6 | FPH5 | FPH4 | FPH3 | FPH2 | FPH1 | FPH0 |
| | | 15:8 | FPH15 | FPH14 | FPH13 | FPH12 | FPH11 | FPH10 | FPH9 | FPH8 |
| | | 23:16 | FPH23 | FPH22 | FPH21 | FPH20 | FPH19 | FPH18 | FPH17 | FPH16 |
| | | 31:24 | FPH31 | FPH30 | FPH29 | FPH28 | FPH27 | FPH26 | FPH25 | FPH24 |
| 0x10C0 | S_PIO_MSKR3 | 7:0 | MSK7 | MSK6 | MSK5 | MSK4 | MSK3 | MSK2 | MSK1 | MSK0 |
| | | 15:8 | MSK15 | MSK14 | MSK13 | MSK12 | MSK11 | MSK10 | MSK9 | MSK8 |
| | | 23:16 | MSK23 | MSK22 | MSK21 | MSK20 | MSK19 | MSK18 | MSK17 | MSK16 |
| | | 31:24 | MSK31 | MSK30 | MSK29 | MSK28 | MSK27 | MSK26 | MSK25 | MSK24 |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x10C4 | S_PIO_CFGR3 | 7:0 | | | | | | FUNC[2:0] | | |
| | | 15:8 | SCHMITT | OPD | IFSCEN | IFEN | | PDEN | PUEN | DIR |
| | | 23:16 | | | | | | | DRVSTR[1:0] | |
| | | 31:24 | | ICFS | PCFS | | | EVTSEL[2:0] | | |
| 0x10C8 | S_PIO_PDSR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10CC | S_PIO_LOCKSR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10D0 | S_PIO_SODR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10D4 | S_PIO_CODR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10D8 | S_PIO_ODSR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10DC ... 0x10DF | Reserved | | | | | | | | | |
| 0x10E0 | S_PIO_IER3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10E4 | S_PIO_IDR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10E8 | S_PIO_IMR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10EC | S_PIO_ISR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x10F0 | S_PIO_SIONR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10F4 | S_PIO_SIOSR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10F8 | S_PIO_IOSSR3 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x10FC | S_PIO_IOFR3 | 7:0 | FPH7 | FPH6 | FPH5 | FPH4 | FPH3 | FPH2 | FPH1 | FPH0 |
| | | 15:8 | FPH15 | FPH14 | FPH13 | FPH12 | FPH11 | FPH10 | FPH9 | FPH8 |
| | | 23:16 | FPH23 | FPH22 | FPH21 | FPH20 | FPH19 | FPH18 | FPH17 | FPH16 |
| | | 31:24 | FPH31 | FPH30 | FPH29 | FPH28 | FPH27 | FPH26 | FPH25 | FPH24 |
| 0x1100 ... 0x14FF | Reserved | | | | | | | | | |
| 0x1500 | S_PIO_SCDR | 7:0 | DIV[7:0] | | | | | | | |
| | | 15:8 | | | DIV[13:8] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1504 ... 0x15DF | Reserved | | | | | | | | | |
| 0x15E0 | S_PIO_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0x15E4 | S_PIO_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |

### 33.7.1 PIO Mask Register

**Name:** PIO_MSKRx
**Offset:** 0x00 + x*0x40 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | MSK31 | MSK30 | MSK29 | MSK28 | MSK27 | MSK26 | MSK25 | MSK24 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MSK23 | MSK22 | MSK21 | MSK20 | MSK19 | MSK18 | MSK17 | MSK16 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MSK15 | MSK14 | MSK13 | MSK12 | MSK11 | MSK10 | MSK9 | MSK8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MSK7 | MSK6 | MSK5 | MSK4 | MSK3 | MSK2 | MSK1 | MSK0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – MSKy** PIO Line y Mask
These bits define the I/O lines to be configured when writing the PIO Configuration Register.

0 (DISABLED): Writing the PIO_CFGRx, PIO_ODSRx or PIO_IOFRx does not affect the corresponding I/O line configuration.

1 (ENABLED): Writing the PIO_CFGRx, PIO_ODSRx or PIO_IOFRx updates the corresponding I/O line configuration.

### 33.7.2 PIO Configuration Register

**Name:** PIO_CFGRx
**Offset:** 0x04 + x*0x40 [x=0..3]
**Reset:** 0x00000200
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

Writing this register will only affect I/O lines enabled in the PIO_MSKRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | ICFS | PCFS | | | EVTSEL[2:0] | | |
| Access | | R | R | | | R/W | R/W | R/W |
| Reset | | 0 | 0 | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | DRVSTR[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SCHMITT | OPD | IFSCEN | IFEN | | PDEN | PUEN | DIR |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 1 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | FUNC[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 30 – ICFS** Interrupt Configuration Freeze Status (read-only)
This bit gives information about the freeze state of the following fields of the read I/O line configuration:

- IFEN: Input Filter Enable
- IFSCEN: Input Filter Slow Clock Enable
- EVTSEL: Event Selection

0 (NOT_FROZEN): The fields are not frozen and can be written for this I/O line.

1 (FROZEN): The fields are frozen and cannot be written for this I/O line. Only a hardware reset can release these fields.

**Bit 29 – PCFS** Physical Configuration Freeze Status (read-only)
This bit gives information about the freeze state of the following fields of the read I/O line configuration:

- FUNC: I/O Line Function
- DIR: Direction
- PUEN: Pull-Up Enable
- PDEN: Pull-Down Enable

• OPD: Open-Drain

• SCHMITT: Schmitt Trigger

• DRVSTR: Drive Strength

0 (NOT_FROZEN): The fields are not frozen and can be written for this I/O line.

1 (FROZEN): The fields are frozen and cannot be written for this I/O line. Only a hardware reset can release these fields.

**Bits 26:24 – EVTSEL[2:0]** Event Selection
This field defines the type of event to detect on the I/O lines of the I/O group x according to the PIO Mask Register.

| Value | Name | Description |
|---|---|---|
| 0 | FALLING | Event detection on input falling edge |
| 1 | RISING | Event detection on input rising edge |
| 2 | BOTH | Event detection on input both edge |
| 3 | LOW | Event detection on low level input |
| 4 | HIGH | Event detection on high level input |
| 5 | – | Reserved |
| 6 | – | Reserved |
| 7 | – | Reserved |

**Bits 17:16 – DRVSTR[1:0]** Drive Strength
This field defines the drive strength of the I/O lines of the I/O group x according to the PIO Mask Register.

| Value | Name | Description |
|---|---|---|
| 0 | LO | Low drive |
| 1 | LO | Low drive |
| 2 | ME | Medium drive |
| 3 | HI | High drive |

**Bit 15 – SCHMITT** Schmitt Trigger
This bit defines the Schmitt trigger configuration of the I/O lines of the I/O group x according to the PIO Mask Register.

0 (ENABLED): Schmitt trigger is enabled for the selected I/O lines.

1 (DISABLED): Schmitt trigger is disabled for the selected I/O lines.

**Bit 14 – OPD** Open-Drain
This bit defines the open drain configuration of the I/O lines of the I/O group x according to the PIO Mask Register.

0 (DISABLED): The open-drain is disabled for the selected I/O lines. I/O lines are driven at high- and low-level.

1 (ENABLED): The open-drain is enabled for the selected I/O lines. I/O lines are driven at low-level only.

**Bit 13 – IFSCEN** Input Filter Slow Clock Enable
This bit defines the clock source of the glitch filtering for the I/O lines of the I/O group x according to the PIO Mask Register.

0 (DISABLED): The glitch filter is able to filter glitches with a duration < $t_{mck}/2$ for the selected I/O lines.

1 (ENABLED): The debouncing filter is able to filter pulses with a duration < $t_{div_slck}/2$ for the selected I/O lines.

**Bit 12 – IFEN** Input Filter Enable
This bit defines if the glitch filtering is used for the I/O lines of the I/O group x according to the PIO Mask Register.

0 (DISABLED): The input filter is disabled for the selected I/O lines.

1 (ENABLED): The input filter is enabled for the selected I/O lines.

**Bit 10 – PDEN** Pull-Down Enable
This bit defines the pull-down configuration of the I/O lines of the I/O group x according to the PIO Mask Register.

PDEN can be written to 1 only if PUEN is written to 0.

0 (DISABLED): Pull-Down is disabled for the selected I/O lines.

1 (ENABLED): Pull-Down is enabled for the selected I/O lines only if PUEN is 0.

**Bit 9 – PUEN** Pull-Up Enable
This bit defines the pull-up configuration of the I/O lines of the I/O group x according to the PIO Mask Register.

0 (DISABLED): Pull-Up is disabled for the selected I/O lines.

1 (ENABLED): Pull-Up is enabled for the selected I/O lines.

**Bit 8 – DIR** Direction
This bit defines the direction of the I/O lines of the I/O group x according to the PIO Mask Register.

0 (INPUT): The selected I/O lines are pure inputs.

1 (OUTPUT): The selected I/O lines are enabled in output.

**Bits 2:0 – FUNC[2:0]** I/O Line Function
This field defines the function for I/O lines of the I/O group x according to the PIO Mask Register.

| Value | Name | Description |
|---|---|---|
| 0 | GPIO | Select the PIO mode for the selected I/O lines. |
| 1 | PERIPH_A | Select the peripheral A for the selected I/O lines. |
| 2 | PERIPH_B | Select the peripheral B for the selected I/O lines. |
| 3 | PERIPH_C | Select the peripheral C for the selected I/O lines. |
| 4 | PERIPH_D | Select the peripheral D for the selected I/O lines. |
| 5 | PERIPH_E | Select the peripheral E for the selected I/O lines. |
| 6 | PERIPH_F | Select the peripheral F for the selected I/O lines. |

### 33.7.3 PIO Pin Data Status Register

**Name:** PIO_PDSRx
**Offset:** 0x08 + x*0x40 [x=0..3]
**Reset:** –
**Property:** Read-only

Reset value of PIO_PDSR depends on the level of the I/O lines. Reading the I/O line levels requires the clock of the PIO Controller to be enabled, otherwise PIO_PDSR reads the levels present on the I/O line at the time the clock was disabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Input Data Status

| Value | Description |
|---|---|
| 0 | The I/O line of the I/O group x is at level 0. |
| 1 | The I/O line of the I/O group x is at level 1. |

### 33.7.4 PIO Lock Status Register

**Name:** PIO_LOCKSRx
**Offset:** 0x0C + x*0x40 [x=0..3]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Lock Status

| Value | Description |
|---|---|
| 0 | The I/O line of the I/O group x is not locked. |
| 1 | The I/O line of the I/O group x is locked. |

### 33.7.5 PIO Set Output Data Register

**Name:** PIO_SODRx
**Offset:** 0x10 + x*0x40 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Set Output Data

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Sets the data to be driven on the I/O line of I/O group x. |

### 33.7.6 PIO Clear Output Data Register

**Name:** PIO_CODRx
**Offset:** 0x14 + x*0x40 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Clear Output Data

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears the data to be driven on the I/O line of the I/O group x. |

### 33.7.7 PIO Output Data Status Register

**Name:** PIO_ODSRx
**Offset:** 0x18 + x*0x40 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

Writing this register will only affect I/O lines enabled in the PIO_MSKRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Output Data Status

| Value | Description |
|---|---|
| 0 | The data to be driven on the I/O line of the I/O group x is 0. |
| 1 | The data to be driven on the I/O line of the I/O group x is 1. |

### 33.7.8 PIO Interrupt Enable Register

**Name:** PIO_IERx
**Offset:** 0x20 + x*0x40 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Input Change Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Input Change interrupt on the I/O line of the I/O group x. |

### 33.7.9 PIO Interrupt Disable Register

**Name:** PIO_IDRx
**Offset:** 0x24 + x*0x40 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Input Change Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Input Change interrupt on the I/O line of the I/O group x. |

### 33.7.10 PIO Interrupt Mask Register

**Name:** PIO_IMRx
**Offset:** 0x28 + x*0x40 [x=0..3]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Input Change Interrupt Mask

| Value | Description |
|---|---|
| 0 | Input Change interrupt is disabled on the I/O line of the I/O group x. |
| 1 | Input Change interrupt is enabled on the I/O line of the I/O group x. |

### 33.7.11 PIO Interrupt Status Register

**Name:** PIO_ISRx
**Offset:** 0x30 + x*0x40 [x=0..3]
**Reset:** 0x00000000
**Property:** Read-only

PIO_ISR is reset at 0x00000000. However, the first read of the register may read a different value as input changes may have occurred.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Input Change Interrupt Status

| Value | Description |
|---|---|
| 0 | No Input Change has been detected on the I/O line of the I/O group x since PIO_ISRx was last read or since reset. |
| 1 | At least one Input Change has been detected on the I/O line of the I/O group since PIO_ISRx was last read or since reset. |

### 33.7.12 PIO I/O Freeze Configuration Register

**Name:** PIO_IOFRx
**Offset:** 0x3C + x*0x40 [x=0..3]
**Reset:** –
**Property:** Write-only

Writing this register will only affect I/O lines enabled in the PIO_MSKRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FRZKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FRZKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FRZKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | FINT | FPHY |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

**Bits 31:8 – FRZKEY[23:0]** Freeze Key

| Value | Name | Description |
|---|---|---|
| 0x494F46 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit. |

**Bit 1 – FINT** Freeze Interrupt Configuration
Only a hardware reset can reset the FINT bit.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Freezes the following configuration fields of Non-Secure I/O lines if FRZKEY corresponds to 0x494F46 (“IOF” in ASCII):<br>• IFEN: Input Filter Enable<br>• IFSCEN: Input Filter Slow Clock Enable<br>• EVTSEL: Event Selection |

**Bit 0 – FPHY** Freeze Physical Configuration
Only a hardware reset can reset the FPHY bit.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Freezes the following configuration fields of Non-Secure I/O lines if FRZKEY corresponds to 0x494F46 (“IOF” in ASCII):<br>• FUNC: I/O Line Function<br>• DIR: Direction<br>• PUEN: Pull-Up Enable<br>• PDEN: Pull-Down Enable<br>• OPD: Open-Drain<br>• SCHMITT: Schmitt Trigger<br>• DRVSTR: Drive Strength |

### 33.7.13 PIO Write Protection Mode Register

**Name:** PIO_WPMR
**Offset:** 0x5E0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x504944F | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See Section 6.18 “Register Write Protection” for the list of registers that can be protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x50494F (“PIO” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x50494F (“PIO” in ASCII). |

### 33.7.14 PIO Write Protection Status Register

**Name:** PIO_WPSR
**Offset:** 0x5E4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of the PIO_WPSR. |
| 1 | A write protection violation has occurred since the last read of the PIO_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

### 33.7.15 Secure PIO Mask Register

**Name:** S_PIO_MSKRx
**Offset:** 0x1000 + x*0x40 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the Secure PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | MSK31 | MSK30 | MSK29 | MSK28 | MSK27 | MSK26 | MSK25 | MSK24 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MSK23 | MSK22 | MSK21 | MSK20 | MSK19 | MSK18 | MSK17 | MSK16 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MSK15 | MSK14 | MSK13 | MSK12 | MSK11 | MSK10 | MSK9 | MSK8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MSK7 | MSK6 | MSK5 | MSK4 | MSK3 | MSK2 | MSK1 | MSK0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – MSKy** PIO Line y Mask
These bits define the I/O lines to be configured when writing the Secure PIO Configuration Register.

0 (DISABLED): Writing the S_PIO_CFGRx, S_PIO_ODSRx or S_PIO_IOFRx does not affect the corresponding I/O line configuration.

1 (ENABLED): Writing the S_PIO_CFGRx, S_PIO_ODSRx or S_PIO_IOFRX updates the corresponding I/O line configuration.

### 33.7.16 Secure PIO Configuration Register

**Name:** S_PIO_CFGRx
**Offset:** 0x1004 + x*0x40 [x=0..3]
**Reset:** 0x00000200
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the Secure PIO Write Protection Mode Register.

Writing this register will only affect I/O lines enabled in the S_PIO_MSKRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | ICFS | PCFS | | | EVTSEL[2:0] | | |
| Access | | R | R | | | R/W | R/W | R/W |
| Reset | | 0 | 0 | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | DRVSTR[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SCHMITT | OPD | IFSCEN | IFEN | | PDEN | PUEN | DIR |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 1 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | FUNC[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 30 – ICFS** Interrupt Configuration Freeze Status (read-only)
This bit gives information about the freeze state of the following fields of the read I/O line configuration:

- IFEN: Input Filter Enable
- IFSCEN: Input Filter Slow Clock Enable
- EVTSEL: Event Selection

0 (NOT_FROZEN): The fields are not frozen and can be written for this I/O line.

1 (FROZEN): The fields are frozen and cannot be written for this I/O line. Only a hardware reset can release these fields.

**Bit 29 – PCFS** Physical Configuration Freeze Status (read-only)
This bit gives information about the freeze state of the following fields of the read I/O line configuration:

- FUNC: I/O Line Function
- DIR: Direction
- PUEN: Pull-Up Enable

• PDEN: Pull-Down Enable

• OPD: Open-Drain

• SCHMITT: Schmitt Trigger

• DRVSTR: Drive Strength

0 (NOT_FROZEN): The fields are not frozen and can be written for this I/O line.

1 (FROZEN): The fields are frozen and cannot be written for this I/O line. Only a hardware reset can release these fields.

**Bits 26:24 – EVTSEL[2:0]** Event Selection
This field defines the type of event to detect on the I/O lines of the I/O group x according to the Secure PIO Mask Register.

| Value | Name | Description |
|---|---|---|
| 0 | FALLING | Event detection on input falling edge |
| 1 | RISING | Event detection on input rising edge |
| 2 | BOTH | Event detection on input both edge |
| 3 | LOW | Event detection on low level input |
| 4 | HIGH | Event detection on high level input |
| 5 | Reserved | |
| 6 | Reserved | |
| 7 | Reserved | |

**Bits 17:16 – DRVSTR[1:0]** Drive Strength
This field defines the drive strength of the I/O lines of the I/O group x according to the Secure PIO Mask Register.

| Value | Name | Description |
|---|---|---|
| 0 | LO | Low drive |
| 1 | LO | Low drive |
| 2 | ME | Medium drive |
| 3 | HI | High drive |

**Bit 15 – SCHMITT** Schmitt Trigger
This bit defines the Schmitt trigger configuration of the I/O lines of the I/O group x according to the Secure PIO Mask Register.

0 (ENABLED): Schmitt trigger is enabled for the selected I/O lines.

1 (DISABLED): Schmitt trigger is disabled for the selected I/O lines.

**Bit 14 – OPD** Open-Drain
This bit defines the open drain configuration of the I/O lines of the I/O group x according to the Secure PIO Mask Register.

0 (DISABLED): The open-drain is disabled for the selected I/O lines. I/O lines are driven at high- and low-level.

1 (ENABLED): The open-drain is enabled for the selected I/O lines. I/O lines are driven at low-level only.

**Bit 13 – IFSCEN** Input Filter Slow Clock Enable
This bit defines the clock source of the glitch filtering for the I/O lines of the I/O group x according to the Secure PIO Mask Register.

| Value | Description |
|---|---|
| 0 | The glitch filter is able to filter glitches with a duration < $t_{mck}/2$ for the selected I/O lines. |
| 1 | The debouncing filter is able to filter pulses with a duration < $t_{div_slck}/2$ for the selected I/O lines. |

**Bit 12 – IFEN** Input Filter Enable
This bit defines if the glitch filtering is used for the I/O lines of the I/O group x according to the Secure PIO Mask Register.

0 (DISABLED): The input filter is disabled for the selected I/O lines.

1 (ENABLED): The input filter is enabled for the selected I/O lines.

**Bit 10 – PDEN** Pull-Down Enable
This bit defines the pull-down configuration of the I/O lines of the I/O group x according to the Secure PIO Mask Register.

0 (DISABLED): Pull-Down is disabled for the selected I/O lines.

1 (ENABLED): Pull-Down is enabled for the selected I/O lines only if PUEN is 0[(1)].

Note: 1. PDEN can be written to 1 only if PUEN is written to 0.

**Bit 9 – PUEN** Pull-Up Enable
This bit defines the pull-up configuration of the I/O lines of the I/O group x according to the Secure PIO Mask Register.

0 (DISABLED): Pull-Up is disabled for the selected I/O lines.

1 (ENABLED): Pull-Up is enabled for the selected I/O lines.

**Bit 8 – DIR** Direction
This bit defines the direction of the I/O lines of the I/O group x according to the Secure PIO Mask Register.

0 (INPUT): The selected I/O lines are pure inputs.

1 (OUTPUT): The selected I/O lines are enabled in output.

**Bits 2:0 – FUNC[2:0]** I/O Line Function
This field defines the function for I/O lines of the I/O group x according to the Secure PIO Mask Register.

| Value | Name | Description |
|---|---|---|
| 0 | GPIO | Select the PIO mode for the selected I/O lines. |
| 1 | PERIPH_A | Select the peripheral A for the selected I/O lines. |
| 2 | PERIPH_B | Select the peripheral B for the selected I/O lines. |
| 3 | PERIPH_C | Select the peripheral C for the selected I/O lines. |
| 4 | PERIPH_D | Select the peripheral D for the selected I/O lines. |
| 5 | PERIPH_E | Select the peripheral E for the selected I/O lines. |
| 6 | PERIPH_F | Select the peripheral F for the selected I/O lines. |
| 7 | PERIPH_G | Select the peripheral G for the selected I/O lines. |

© 2018 Microchip Technology Inc.

### 33.7.17 Secure PIO Pin Data Status Register

**Name:** S_PIO_PDSRx
**Offset:** 0x1008 + x*0x40 [x=0..3]
**Reset:** –
**Property:** Read-only

Reset value of PIO_PDSR and S_PIO_PDSR depends on the level of the I/O lines. Reading the I/O line levels requires the clock of the PIO Controller to be enabled, otherwise PIO_PDSR reads the levels present on the I/O line at the time the clock was disabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Input Data Status

| Value | Description |
|---|---|
| 0 | The I/O line of the I/O group x is at level 0. |
| 1 | The I/O line of the I/O group x is at level 1. |

### 33.7.18 Secure PIO Lock Status Register

**Name:** S_PIO_LOCKSRx
**Offset:** 0x100C + x*0x40 [x=0..3]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Lock Status

| Value | Description |
|---|---|
| 0 | The I/O line of the I/O group x is not locked. |
| 1 | The I/O line of the I/O group x is locked. |

### 33.7.19 Secure PIO Set Output Data Register

**Name:** S_PIO_SODRx
**Offset:** 0x1010 + x*0x40 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Set Output Data

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Sets the data to be driven on the I/O line of I/O group x. |

### 33.7.20 Secure PIO Clear Output Data Register

**Name:** S_PIO_CODRx
**Offset:** 0x1014 + x*0x40 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Clear Output Data

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears the data to be driven on the I/O line of the I/O group x. |

### 33.7.21 Secure PIO Output Data Status Register

**Name:** S_PIO_ODSRx
**Offset:** 0x1018 + x*0x40 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

Writing this register will only affect I/O lines enabled in the S_PIO_MSKRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Output Data Status

| Value | Description |
|---|---|
| 0 | The data to be driven on the I/O line of the I/O group x is 0. |
| 1 | The data to be driven on the I/O line of the I/O group x is 1. |

### 33.7.22 Secure PIO Interrupt Enable Register

**Name:** S_PIO_IERx
**Offset:** 0x1020 + x*0x40 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Input Change Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Input Change interrupt on the I/O line of the I/O group x. |

### 33.7.23 Secure PIO Interrupt Disable Register

**Name:** S_PIO_IDRx
**Offset:** 0x1024 + x*0x40 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Input Change Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Input Change interrupt on the I/O line of the I/O group x. |

### 33.7.24 Secure PIO Interrupt Mask Register

**Name:** S_PIO_IMRx
**Offset:** 0x1028 + x*0x40 [x=0..3]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Input Change Interrupt Mask

| Value | Description |
|---|---|
| 0 | Input Change interrupt is disabled on the I/O line of the I/O group x. |
| 1 | Input Change interrupt is enabled on the I/O line of the I/O group x. |

### 33.7.25 Secure PIO Interrupt Status Register

**Name:** S_PIO_ISRx
**Offset:** 0x102C + x*0x40 [x=0..3]
**Reset:** 0x00000000
**Property:** Read-only

PIO_ISR and S_PIO_ISR are reset at 0x000000000. However, the first read of the register may read a different value as input changes may have occurred.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Input Change Interrupt Status

| Value | Description |
|---|---|
| 0 | No Input Change has been detected on the I/O line of the I/O group x since S_PIO_ISRx was last read or since reset. |
| 1 | At least one Input Change has been detected on the I/O line of the I/O group since S_PIO_ISRx was last read or since reset. |

### 33.7.26 Secure PIO Set I/O Non-Secure Register

**Name:** S_PIO_SIONRx
**Offset:** 0x1030 + x*0x40 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Set I/O Non-Secure

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Set the I/O line of the I/O group x in Non-Secure mode. |

### 33.7.27 Secure PIO Set I/O Secure Register

**Name:** S_PIO_SIOSRx
**Offset:** 0x1034 + x*0x40 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** Set I/O Secure

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Set the I/O line of the I/O group x in Secure mode. |

### 33.7.28 Secure PIO I/O Security Status Register

**Name:** S_PIO_IOSSRx
**Offset:** 0x1038 + x*0x40 [x=0..3]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – Px** I/O Security Status
0 (SECURE): The I/O line of the I/O group x is in Secure mode.

1 (NON_SECURE): The I/O line of the I/O group x is in Non-Secure mode.

### 33.7.29 Secure PIO I/O Freeze Configuration Register

**Name:** S_PIO_IOFRx
**Offset:** 0x103C + x*0x40 [x=0..3]
**Reset:** –
**Property:** Write-only

Writing this register will only affect I/O lines enabled in the S_PIO_MSKRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FPH31 | FPH30 | FPH29 | FPH28 | FPH27 | FPH26 | FPH25 | FPH24 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FPH23 | FPH22 | FPH21 | FPH20 | FPH19 | FPH18 | FPH17 | FPH16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FPH15 | FPH14 | FPH13 | FPH12 | FPH11 | FPH10 | FPH9 | FPH8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FPH7 | FPH6 | FPH5 | FPH4 | FPH3 | FPH2 | FPH1 | FPH0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 31:8 – FRZKEY[23:0]** Freeze Key

| Value | Name | Description |
|---|---|---|
| 0x494F46 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit. |

**Bit 1 – FINT** Freeze Interrupt Configuration
Only a hardware reset can reset the FINT bit.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Freezes the following configuration fields of Secure I/O lines if FRZKEY corresponds to 0x494F46 (“IOF” in ASCII):<br>• IFEN: Input Filter Enable<br>• IFSCEN: Input Filter Slow Clock Enable<br>• EVTSEL: Event Selection |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – FPHY** Freeze Physical Configuration
Only a hardware reset can reset the FPHY bit.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Freezes the following configuration fields of Secure I/O lines if FRZKEY corresponds to 0x494F46 (“IOF” in ASCII):<br>• FUNC: I/O Line Function<br>• DIR: Direction<br>• PUEN: Pull-Up Enable<br>• PDEN: Pull-Down Enable<br>• OPD: Open-Drain<br>• SCHMITT: Schmitt Trigger<br>• DRVSTR: Drive Strength |

### 33.7.30 Secure PIO Slow Clock Divider Debouncing Register

**Name:** S_PIO_SCDR
**Offset:** 0x1500
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the Secure PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | DIV[13:8] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DIV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 13:0 – DIV[13:0]** Slow Clock Divider Selection for Debouncing
$t_{div_slck} = ((DIV + 1) \times 2) \times t_{slck}$

### 33.7.31 Secure PIO Write Protection Mode Register

**Name:** S_PIO_WPMR
**Offset:** 0x15E0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x504944F | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See Register Write Protection for the list of registers that can be protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x50494F (“PIO” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x50494F (“PIO” in ASCII). |

### 33.7.32 Secure PIO Write Protection Status Register

**Name:** S_PIO_WPSR
**Offset:** 0x15E4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of the S_PIO_WPSR. |
| 1 | A write protection violation has occurred since the last read of the S_PIO_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 34. External Memories

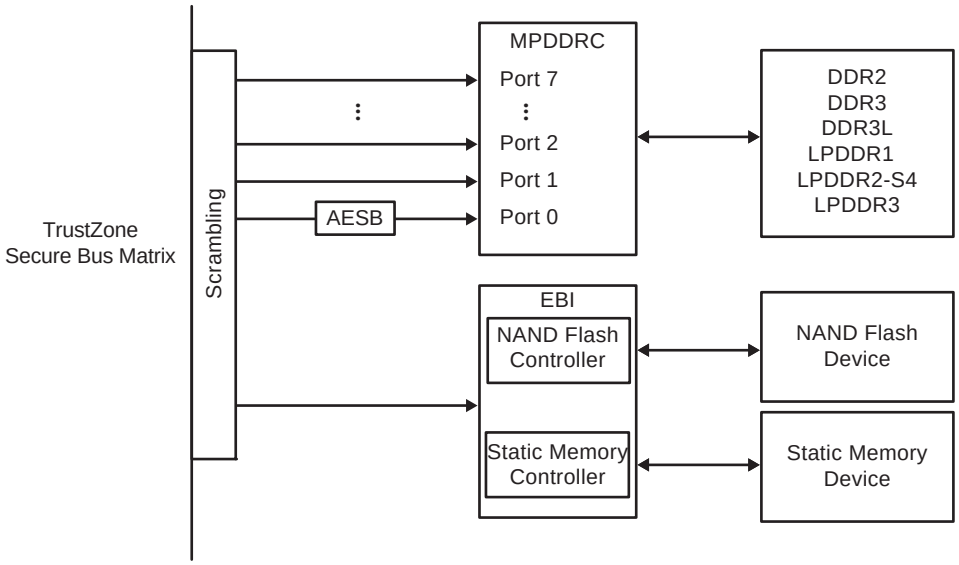
The product features:

- Multiport DDR-SDRAM Controller (MPDDRC)
- External Bus Interface (EBI) that embeds a NAND Flash controller and a Static Memory Controller (HSMC)

The MPDDRC is a multiport DDRSDR controller supporting DDR2, DDR3, DDR3L, LPDDR1, LPDDR2-S4 and LPDDR3 devices. The MPDDRC user interface is located at 0xF000C000. All the paths can be scrambled and Port 0 can be connected to an AES encryption/decryption engine.

The HSMC supports Static Memories and MLC/SLC NAND Flash. It embeds MultiBit ECC correction (PMECC). Its user interface is located at 0xF8014000. The HSMC buses can be scrambled.

**Figure 34-1. External Memory Controllers**




## 34.1 Multiport DDR-SDRAM Controller (MPDDRC)

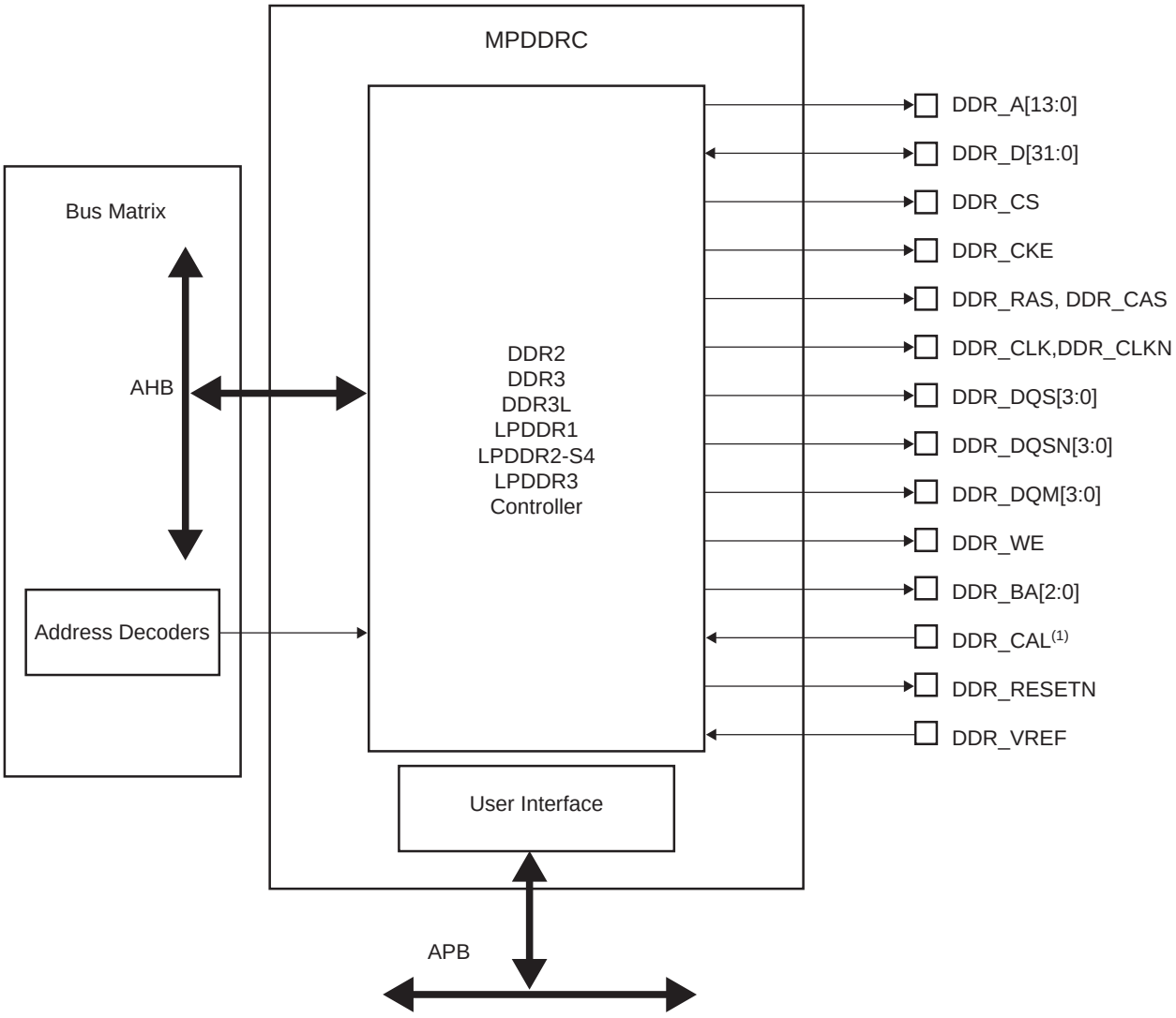
### 34.1.1 Description

The MPDDRC is an 8-port memory controller supporting DDR-SDRAM and low-power DDR devices. Data transfers are performed through a 16/32-bit data bus on one chip select. The controller operates with a 1.8V power supply for DDR2, DDR3, LPDDR1, 1.35V for DDR3L and 1.2V for LPDDR2, LPDDR3.

For full details, refer to the section “Multiport DDR-SDRAM Controller (MPDDRC)”.

### 34.1.2 MPDDR Controller Block Diagram

**Figure 34-2. MPDDRC Block Diagram**




**Note:** For more details, refer to the section ”DDR and SDMMC I/Os Calibration”.

### 34.1.3 IO Lines Description

**Table 34-1. DDR/LPDDR I/O Lines Description**

| Name | Function | Type | Active Level |
|---|---|---|---|
| **DDR/LPDDR Controller** | | | |
| VDDIODDR | Power Supply of memory interface | Power | – |
| DDR_VREF | Reference Voltage | Input | – |
| DDR_CAL | Calibration reference | Input | – |
| DDR_D[31:0] | Data Bus | I/O | – |
| DDR_A[13:0] | Address Bus | Output | – |

.....continued

| Name | Function | Type | Active Level |
|---|---|---|---|
| DDR/LPDDR Controller | | | |
| DDR_DQM[3:0] | Data Mask | Output | – |
| DDR_DQS[3:0] | Data Strobe | I/O | – |
| DDR_DQSN[3:0] | Negative Data Strobe | I/O | – |
| DDR_CS | Chip Select | Output | Low |
| DDR_RESETN | DDR3 Active Low Asynchronous Reset | Output | Low |
| DDR_CLK, DDR_CLKN | Differential Clock | Output | – |
| DDR_CKE | Clock enable | Output | High |
| DDR_RAS | Row signal | Output | Low |
| DDR_CAS | Column signal | Output | Low |
| DDR_WE | Write enable | Output | Low |
| DDR_BA[2:0] | Bank Select | Output | – |

### 34.1.4 Product Dependencies

The pins used for interfacing the DDR/LPDDR memories are not multiplexed with the PIO lines.

The table below gives the connections to the various memory types.

**Table 34-2. I/O Lines Usage vs. Operating Mode**

| Signal Name | DDR2 | DDR3 | DDR3L | LPDDR1 | LPDDR2/ LPDDR3 |
|---|---|---|---|---|---|
| DDR_VREF | VDDIODDR/2 | VDDIODDR/2 | VDDIODDR/2 | VDDIODDR/2 | VDDIODDR/2 |
| DDR_CAL | GND via 21KW | GND via 22KW | GND via 23KW | GND via 21KW | GND via 24KW |
| DDR_CK, DDR_CLKN | CLK, CLKN | CLK, CLKN | CLK, CLKN | CLK, CLKN | CLK, CLKN |
| DDR_CKE | CLKE | CLKE | CLKE | CLKE | CLKE |
| DDR_CS | CS | CS | CS | CS | CS |
| DDR_RESETN | Not connected | DDR_RESETN | DDR_RESETN | Not connected | Not connected |
| DDR_BA[2:0] | BA[2:0] | BA[2:0] | BA[2:0] | BA[1:0] | Not connected |
| DDR_WE | WE | WE | WE | WE | CA2 |
| DDR_RAS, DDR_CAS | RAS, CAS | RAS, CAS | RAS, CAS | RAS, CAS | CA0, CA1 |
| DDR_A[13:0] | A[13:0] | A[13:0] | A[13:0] | A[13:0] | CAx, with x>2 |

| .....continued | | | | | |
|---|---|---|---|---|---|
| **Signal Name** | **DDR2** | **DDR3** | **DDR3L** | **LPDDR1** | **LPDDR2/ LPDDR3** |
| DDR_D[31:0] | D[31:0] | D[31:0] | D[31:0] | D[31:0] | D[31:0] |
| DDR_DQS[3:0], DDR_DQSN[3:0] | LDQS,UDQS, DDR_VREF(1) | DQS[3:0], DQSN[3:0] | DQS[3:0], DQSN[3:0] | DQS[3:0], DDR_VREF | DQS[3:0], DQSN[3:0] |
| DDR_DQM[3:0] | UDM, LDM | DQM[3:0] | DQM[3:0] | DQM[3:0] | DQM[3:0] |

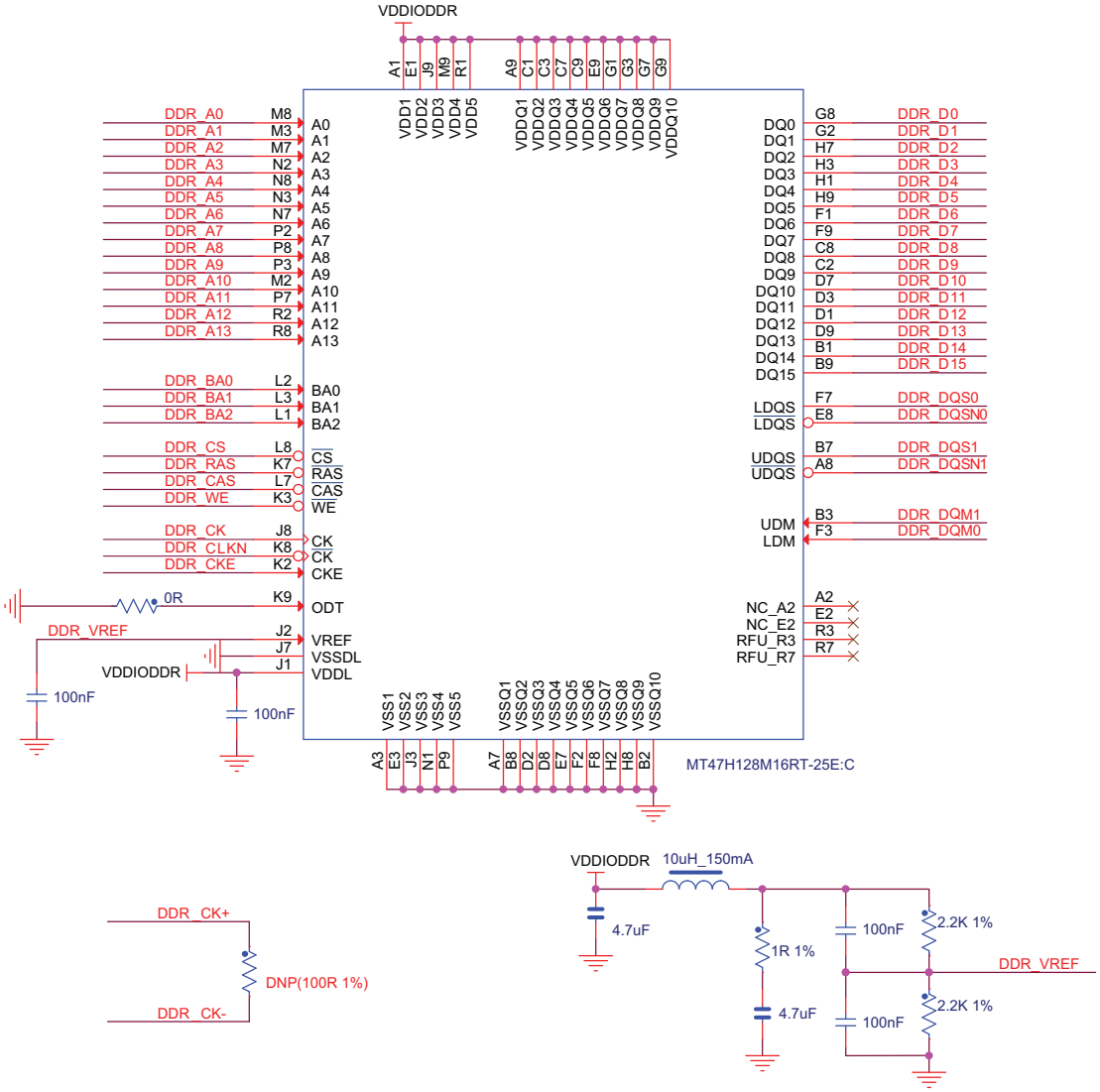
**Note:**

1. DDR_DQSN[3:0] can be connected to DDR_VREF or to DQSN[3:0] of the memory, as specified in the DDR2-SDRAM device datasheet. Bit MPDDRC_CR.NDQS is to be set accordingly.

### 34.1.5 Implementation Examples

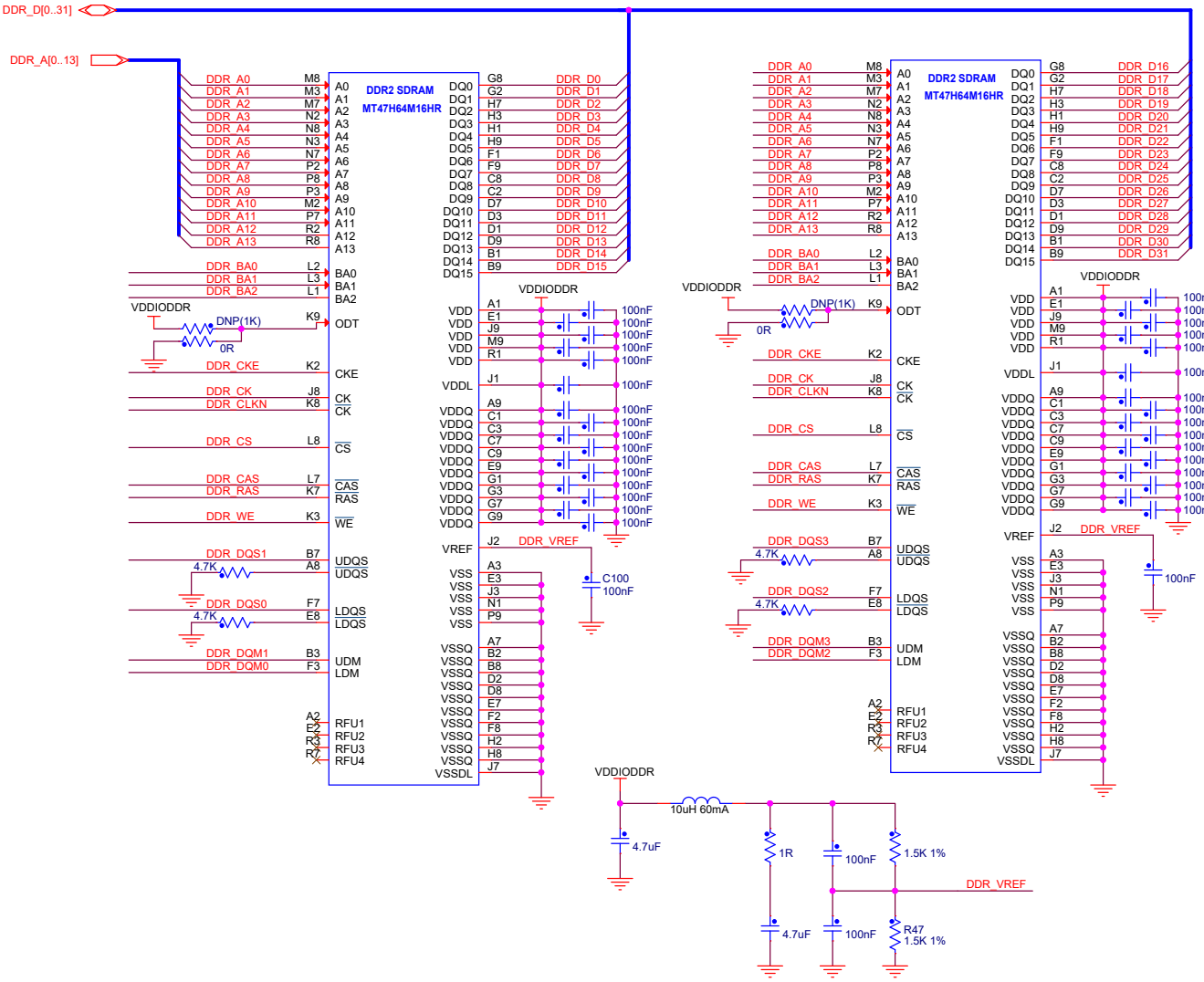
#### 34.1.5.1 16-bit DDR2

**Figure 34-3. 16-bit DDR2 Hardware Configuration**



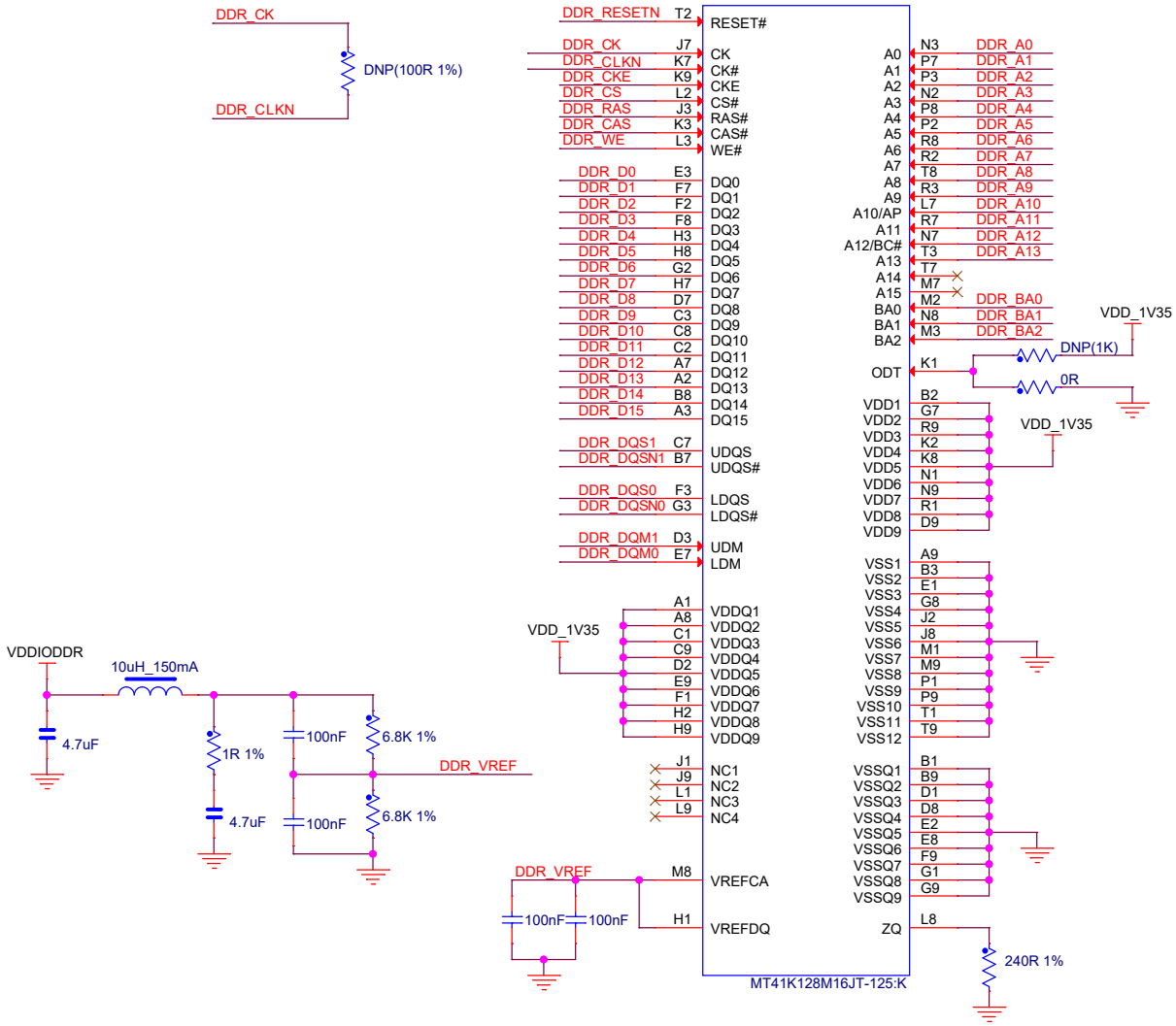
### 34.1.5.2 2x16-bit DDR2

**Figure 34-4. 2x16-bit DDR2 Hardware Configuration**



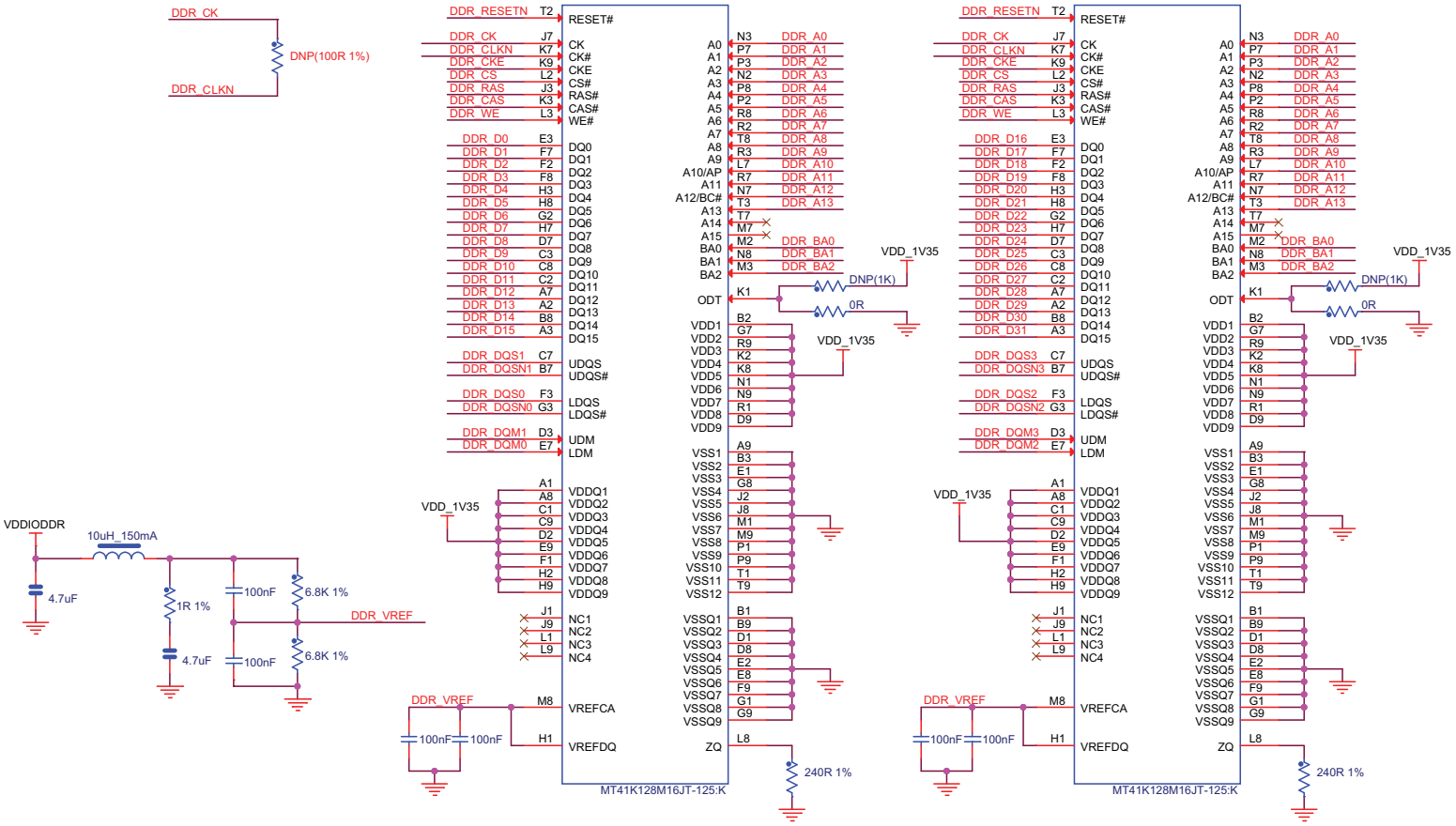
### 34.1.5.3 16-bit DDR3/DDR3L

**Figure 34-5. 16-bit DDR3/DDR3L Hardware Configuration**



### 34.1.5.4 2x16-bit DDR3/DDR3L

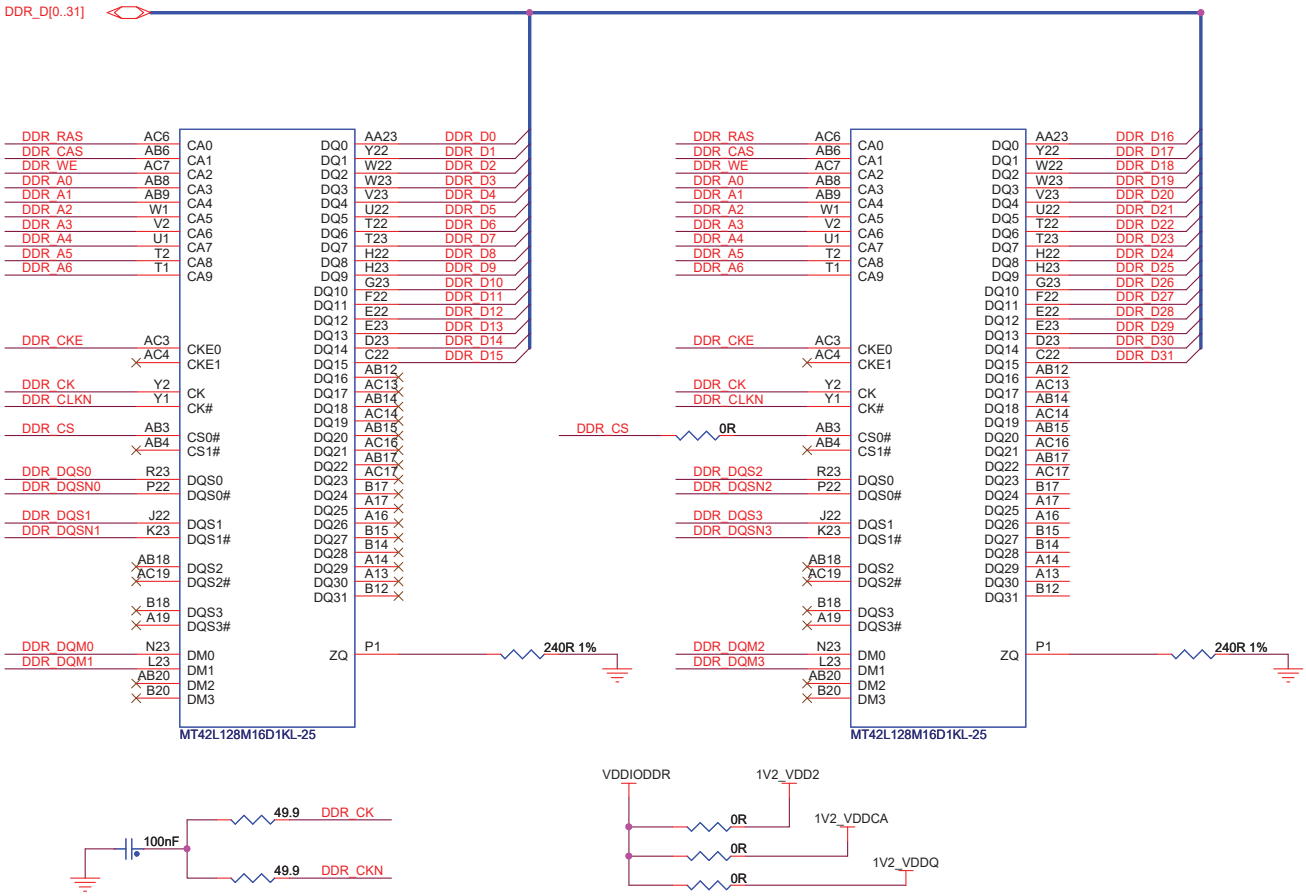
**Figure 34-6. 2x16-bit DDR3/DDR3L Hardware Configuration**




### 34.1.5.5 2x16-bit LPDDR2/LPDDR3

The schematic below is given for LPDDR2 but it is also valid for LPDDR3.

**Figure 34-7. 2x16-bit LPDDR2 Hardware Configuration**




CAx LPDDR2/LPDDR3 signals are to be connected as indicated in the table below.

**Table 34-3. CAx LPDDR2 Signal Connection**

| DDR Controller Signal | LPDDR2 Signal |
|---|---|
| RAS | CA0 |
| CAS | CA1 |
| WE | CA2 |
| DDR_A0 | CA3 |
| DDR_A1 | CA4 |
| DDR_A2 | CA5 |
| DDR_A3 | CA6 |
| DDR_A4 | CA7 |
| DDR_A5 | CA8 |
| DDR_A6 | CA9 |
| Higher addresses | Higher CAs |

## 34.2 External Bus Interface (EBI)

### 34.2.1 Description

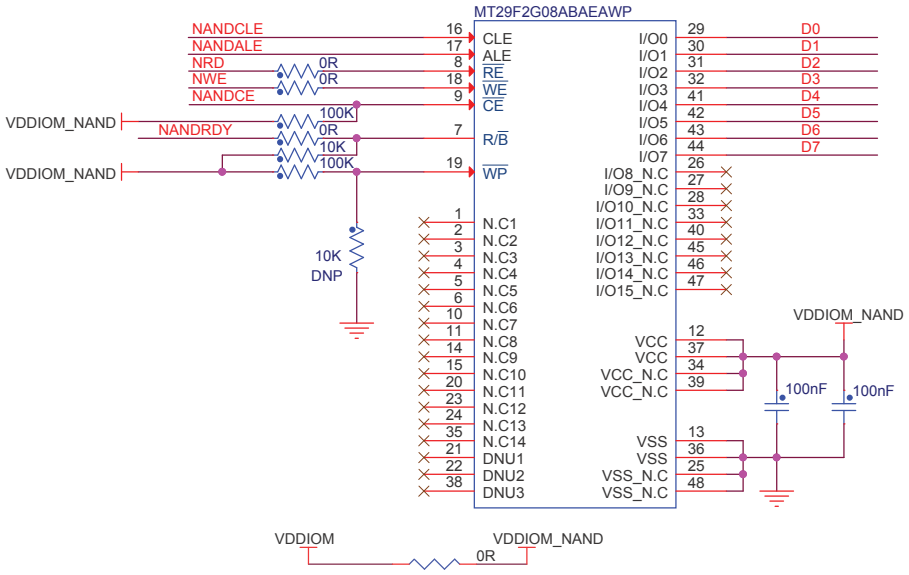
The External Bus Interface is designed to ensure the successful data transfer between several external devices and the Arm processor-based device. The External Bus Interface of the device consists of a Static Memory Controller (SMC).

### 34.2.2 Implementation Examples

The following hardware configurations are given for illustration only. The user should refer to the memory manufacturer website to check current device availability.

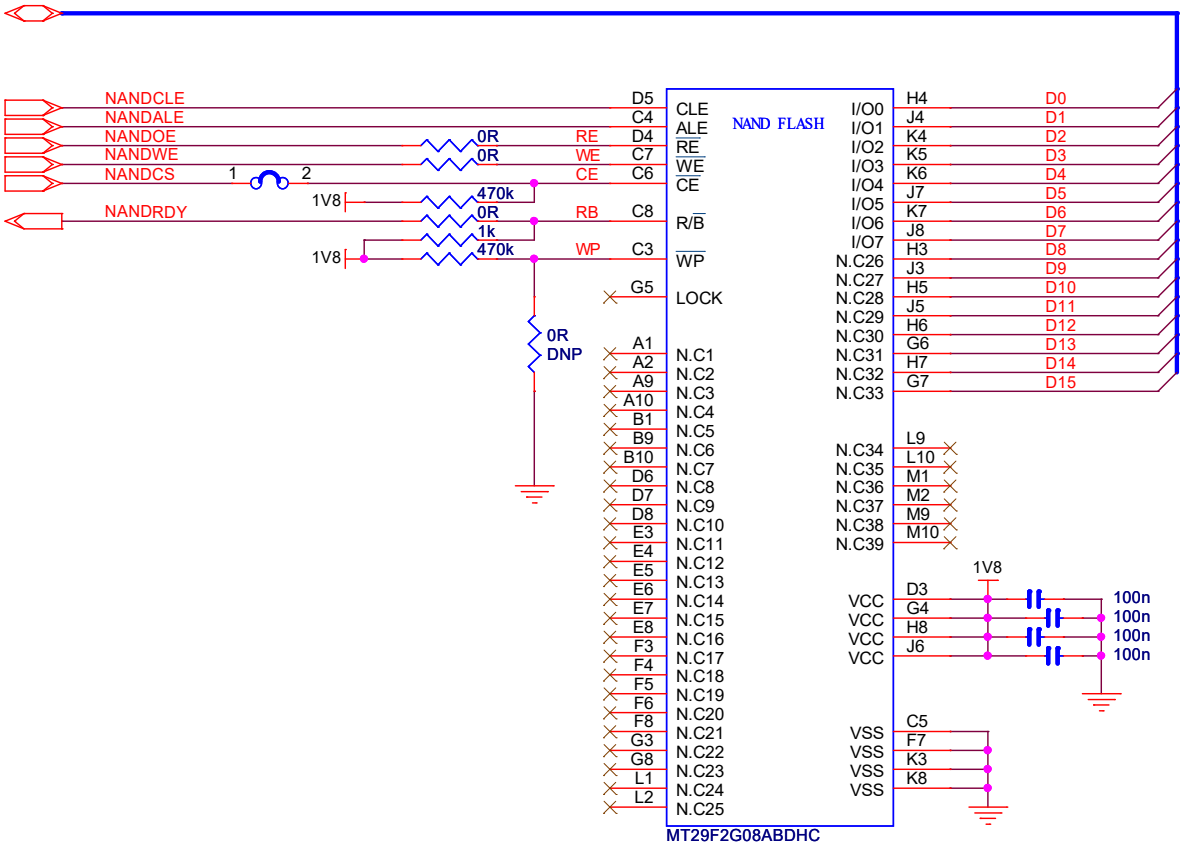
#### 34.2.2.1 8-bit NAND Flash

**Figure 34-8. 8-bit NAND Flash Hardware Configuration**



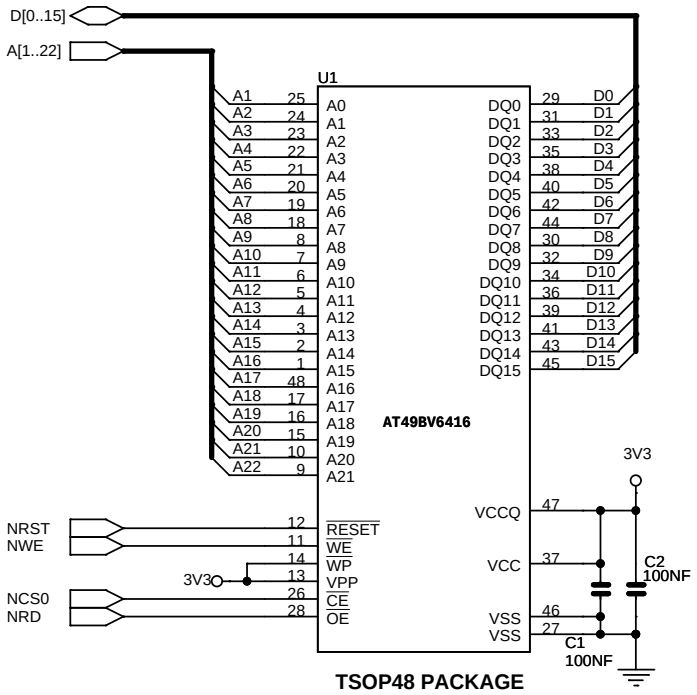
### 34.2.2.2 16-bit NAND Flash

**Figure 34-9. 16-bit NAND Flash Hardware Configuration**



### 34.2.2.3 NOR Flash on NCS0

**Figure 34-10. NOR Flash on NCS0 Hardware Configuration**



# 35. AHB Multiport DDR-SDRAM Controller (MPDDRC)

## 35.1 Description

The Multiport DDR-SDRAM Controller (MPDDRC) is a multiport memory controller. It comprises eight slave AHB interfaces. All simultaneous accesses (eight independent AHB ports) are interleaved to maximize memory bandwidth and minimize transaction latency due to DDR-SDRAM protocol.

The MPDDRC extends the memory capabilities of a chip by providing the interface to the external 16-bit or 32-bit DDR-SDRAM device. The page size supports ranges from 2048 to 16384 rows and from 256 to 4096 columns. It supports dword (64-bit), word (32-bit), half-word (16-bit), and byte (8-bit) accesses.

The MPDDRC supports a read or write burst length of eight locations. This enables the command and address bus to anticipate the next command, thus reducing latency imposed by the DDR-SDRAM protocol and improving the DDR-SDRAM bandwidth. Moreover, MPDDRC keeps track of the active row in each bank, thus maximizing DDR-SDRAM performance, e.g., the application may be placed in one bank and data in other banks. To optimize performance, avoid accessing different rows in the same bank. The MPDDRC supports a CAS latency of 2, 3, 5 or 6 and optimizes the read access depending on the frequency.

Self-refresh, Powerdown and Deep Powerdown modes minimize the consumption of the DDR-SDRAM device.

OCD (Off-chip Driver) and ODT (On-die Termination) modes are not supported.

The MPDDRC supports DDR3-SDRAM and DDR3L-SDRAM devices with DLL disabled, in DLL Off mode. In this mode, according to JEDEC standard, the maximum clock frequency is 125 MHz. However, check with memory suppliers for higher speed support. DDR3-SDRAM supports high capacity, 1 Gbit and more, and allows to reduce power consumption with a 1.5V supply (DDR3-SDRAM) or a 1.35V supply (DDR3L-SDRAM). The DLL Off mode sets the CAS Read Latency (CRL) and the CAS Write Latency (CWL) to 6. The latency is automatically set by the controller.

## 35.2 Embedded Characteristics

- Numerous Memory Devices Supported
  - Low-Power DDR1-SDRAM (LPDDR1)
  - Low-Cost LPDDR1 with 2 Internal Banks
  - DDR2-SDRAM
  - Low-Power DDR2-SDRAM-S4 (LPDDR2)
  - Low-Power DDR3-SDRAM (LPDDR3)
  - DDR3-SDRAM (DLL Off mode)
  - DDR3L-SDRAM (DLL Off mode)
- Arbitration Policies: Round-Robin, On Request, Bandwidth
- Eight Advanced High-Performance Bus (AHB) Interfaces, Management of all Accesses Maximizes Memory Bandwidth and Minimizes Transaction Latency
- Bus Transfer: dword, word, half word, byte Access
- Numerous Configurations Supported
  - 2K, 4K, 8K, 16K row address memory parts

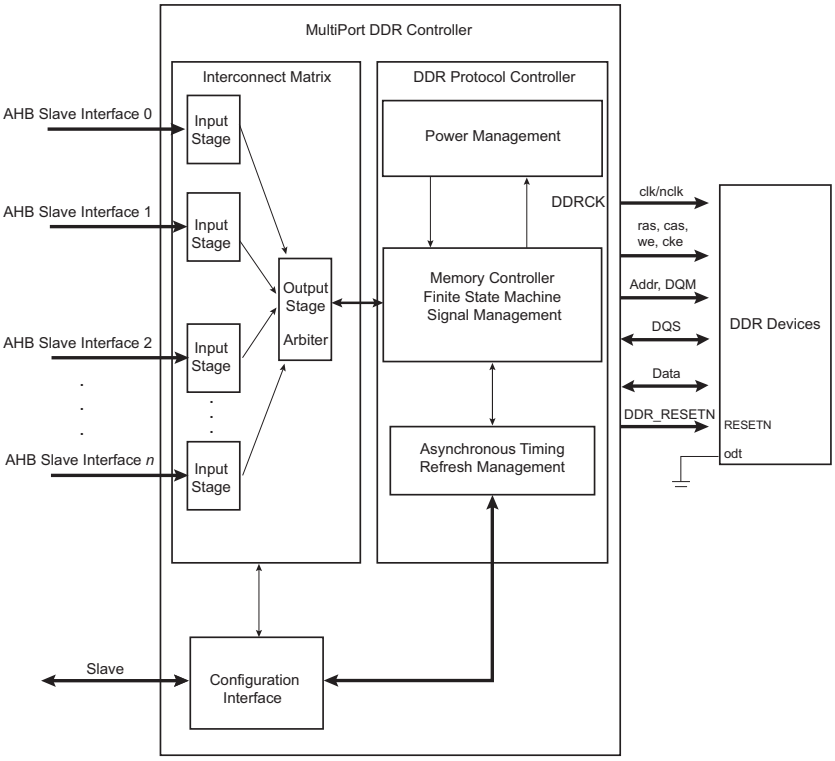
  - DDR-SDRAM with two or four internal banks (low-power DDR1-SDRAM)
  - DDR-SDRAM with four or eight internal banks (DDR2-SDRAM/Low-power DDR2-SDRAM-S4/ DDR3-SDRAM/DDR3L-SDRAM/Low-power DDR3-SDRAM)
  - DDR-SDRAM with 16-bit or 32-bit data
  - One chip select for SDRAM device (512-Mbyte address space, 256-Mbyte address space with 16-bit data path)
- Programming Facilities
  - Multibank ping-pong access (up to four or eight banks opened at the same time = reduced average latency of transactions)
  - Timing parameters specified by software
  - Automatic refresh operation, refresh rate is programmable
  - Automatic update of DS, TCR and PASR parameters (low-power DDR-SDRAM devices)
- Energy-Saving Capabilities
  - Self-refresh, Powerdown, Active Powerdown and Deep Powerdown modes supported
- DDR-SDRAM Powerup Initialization by Software
- CAS Latency of 2, 3, 5 or 6 Supported
- Reset Function Supported (DDR2-SDRAM)
- Clock Frequency Change in Self-Refresh Mode Supported (Low-power DDR-SDRAM/DDR3-SDRAM/DDR3L-SDRAM)
- Autorefresh per Bank Supported (Low-Power DDR2-SDRAM-S4/Low-Power DDR3-SDRAM)
- Automatic Adjust Refresh Rate (Low-Power DDR2-SDRAM-S4/Low-Power DDR3-SDRAM)
- Autoprecharge Command Not Used
- OCD (Off-chip Driver) Mode, ODT (On-die Termination), Write leveling Are Not Supported
- Dynamic Scrambling with User Key (No Impact on Bandwidth)

## 35.3 Block Diagram

The MPDDRC is partitioned in two blocks (see figure below):

- Interconnect Matrix block that manages concurrent accesses on the AHB bus between four AHB masters and integrates an arbiter
- DDR controller that translates AHB requests (read/write) in the DDR-SDRAM protocol

**Figure 35-1. Block Diagram**




## 35.4 Product Dependencies, Initialization Sequence

### 35.4.1 Low-power DDR1-SDRAM Initialization

The initialization sequence is generated by software.

The low-power DDR1-SDRAM devices are initialized by the following sequence:

1. Program the memory device type in the Memory Device register (MPDDRC_MD).
2. Program the shift sampling value in the Read Data Path register (MPDDRC_RD_DATA_PATH).
3. Program the features of the low-power DDR1-SDRAM device in the MPDDRC Configuration register (MPDDRC_CR) (number of columns, rows, banks, CAS latency and output drive strength) and in the Timing Parameter 0 register/Timing Parameter 1 register (MPDDRC_TPR0/1) (asynchronous timing (TRC, TRAS, etc.)).
4. Program Temperature Compensated Self-refresh (TCR), Partial Array Self-refresh (PASR) and Drive Strength (DS) parameters in the Low-power register (MPDDRC_LPR).
5. A NOP command is issued to the low-power DDR1-SDRAM. Program the NOP command in the Mode register (MPDDRC_MR). The application must configure the MODE field to 1 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR1-SDRAM address to acknowledge this command. The clocks which drive the low-power DDR1-SDRAM device are now enabled.
6. A pause of at least 200 µs must be observed before a signal toggle.
7. A NOP command is issued to the low-power DDR1-SDRAM. Program the NOP command in the MPDDRC_MR. The application must configure the MODE field to 1 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write

access to any low-power DDR1-SDRAM address to acknowledge this command. A calibration request is now made to the I/O pad.

8. An All Banks Precharge command is issued to the low-power DDR1-SDRAM. Program All Banks Precharge command in the MPDDRC_MR. The application must configure the MODE field to 2 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR1-SDRAM address to acknowledge this command.
9. Two autorefresh (CBR) cycles are provided. Program the Autorefresh command (CBR) in the MPDDRC_MR. The application must configure the MODE field to 4 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR1-SDRAM location twice to acknowledge these commands.
10. An Extended Mode Register Set (EMRS) cycle is issued to program the low-power DDR1-SDRAM parameters (TCSR, PASR, DS). The application must configure the MODE field to 5 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 1 and BA[0] is set to 0. For example: with a 16-bit, 128-Mbit, low-power DDR1-SDRAM (12 rows, 9 columns, 4 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00800000; with a 32-bit, 1-Gbit, low-power DDR1-SDRAM (14 rows, 10 columns, 4 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x08000000. In the case of low-cost and low-density low-power DDR1-SDRAM (2 internal banks), the write address must be chosen so that signal BA[0] is set to 1. BA[1] is not used.
    **Note:** This address is given as an example only. The real address depends on implementation in the product.
11. A Mode Register Set (MRS) cycle is issued to program parameters of the low-power DDR1-SDRAM devices, in particular CAS latency. The application must configure the MODE field to 3 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the low-power DDR1-SDRAM to acknowledge this command. The write address must be chosen so that signals BA[1:0] are set to 0. For example, the SDRAM write access should be done at the address: BASE_ADDRESS_DDR.
12. The application must enter Normal mode, write a zero to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access at any location in the low-power DDR1-SDRAM to acknowledge this command.
13. Write the refresh rate into the COUNT field in the Refresh Timer register (MPDDRC_RTR). To compute the value, see MPDDRC Refresh Timer Register.

After initialization, the low-power DDR1-SDRAM device is fully functional.

### 35.4.2 DDR2-SDRAM Initialization

The initialization sequence is generated by software. The DDR2-SDRAM devices are initialized by the following sequence:

1. Program the memory device type in the Memory Device register (MPDDRC_MD).
2. Program the shift sampling value in the Read Data Path register (MPDDRC_RD_DATA_PATH).
3. Program features of the DDR2-SDRAM device in the Configuration register (MPDDRC_CR) (number of columns, rows, banks, CAS latency and output driver impedance control) and in the Timing Parameter 0 register/Timing Parameter 1 register (MPDDRC_TPR0/1) (asynchronous timing: TRC, TRAS, etc.).

4. A NOP command is issued to the DDR2-SDRAM. Program the NOP command in the Mode register (MPDDRC_MR). The application must configure the MODE field to 1 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM address to acknowledge this command. The clocks which drive the DDR2-SDRAM device are now enabled.
5. A pause of at least 200 µs must be observed before a signal toggle.
6. A NOP command is issued to the DDR2-SDRAM. Program the NOP command in the MPDDRC_MR. The application must configure the MODE field to 1 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM address to acknowledge this command. CKE is now driven high.
7. An All Banks Precharge command is issued to the DDR2-SDRAM. Program All Banks Precharge command in the MPDDRC_MR. The application must configure the MODE field to 2 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM address to acknowledge this command.
8. An Extended Mode Register Set (EMRS2) cycle is issued to choose between commercial or high temperature operations. The application must configure the MODE field to 5 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 1 and signal BA[0] is set to 0. For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks), the DDR2-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00800000; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x08000000.
   **Note:** This address is given as an example only. The real address depends on implementation in the product.
9. An Extended Mode Register Set (EMRS3) cycle is issued to set the Extended Mode register to 0. The application must configure the MODE field to 5 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 1 and signal BA[0] is set to 1. For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks), the DDR2-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00C00000; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x0C000000
10. An Extended Mode Register Set (EMRS1) cycle is issued to enable DLL and to program D.I.C. (Output Driver Impedance Control). The application must configure the MODE field to 5 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 0 and signal BA[0] is set to 1. For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks), the DDR2-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00400000; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x04000000.
11. An additional 200 cycles of clock are required for locking DLL.
12. Write a ‘1’ to the DLL bit (enable DLL reset) in the Configuration register (MPDDRC_CR).
13. A Mode Register Set (MRS) cycle is issued to reset DLL. The application must configure the MODE field to 3 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this

command. The write address must be chosen so that signals BA[1:0] are set to 0. For example, the SDRAM write access should be done at the address: BASE_ADDRESS_DDR.

14. An All Banks Precharge command is issued to the DDR2-SDRAM. Program the All Banks Precharge command in the MPDDRC_MR. The application must configure the MODE field to 2 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM address to acknowledge this command.
15. Two autorefresh (CBR) cycles are provided. Program the Autorefresh command (CBR) in the MPDDRC_MR. The application must configure the MODE field to 4 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM location twice to acknowledge these commands.
16. Write a '0' to the DLL bit (disable DLL reset) in the MPDDRC_CR.
17. A Mode Register Set (MRS) cycle is issued to program parameters of the DDR2-SDRAM device, in particular CAS latency and to disable DLL reset. The application must configure the MODE field to 3 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signals BA[1:0] are set to 0. For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks) bank address, the SDRAM write access should be done at the address: BASE_ADDRESS_DDR; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR.
18. Configure the OCD field (default OCD calibration) to 7 in the MPDDRC_CR.
19. An Extended Mode Register Set (EMRS1) cycle is issued to the default OCD value. The application must configure the MODE field to 5 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 0 and signal BA[0] is set to 1. For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks), the DDR2-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00400000; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x04000000.
20. Configure the OCD field (exit OCD calibration mode) to 0 in the MPDDRC_CR.
21. An Extended Mode Register Set (EMRS1) cycle is issued to enable OCD exit. The application must configure the MODE field to 5 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 0 and signal BA[0] is set to 1. For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks) bank address, the DDR2-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00400000; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x04000000.
22. A Normal Mode command is provided. Program the Normal mode in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM address to acknowledge this command.
23. Write the refresh rate into the COUNT field in the Refresh Timer register (MPDDRC_RTR). To compute the value, see MPDDRC Refresh Timer Register.

After initialization, the DDR2-SDRAM devices are fully functional.

### 35.4.3 Low-power DDR2-SDRAM Initialization

The initialization sequence is generated by software.

The low-power DDR2-SDRAM devices are initialized by the following sequence:

1. Program the memory device type in the Memory Device register (MPDDRC_MD).
2. Program the shift sampling value in the Read Data Path register (MPDDRC_RD_DATA_PATH).
3. Program features of the low-power DDR2-SDRAM device into and in the Configuration register (MPDDRC_CR) (number of columns, rows, banks, CAS latency and output drive strength) and in the Timing Parameter 0 register/Timing Parameter 1 register (MPDDRC_TPR0/1) (asynchronous timing: TRC, TRAS, etc.).
4. A NOP command is issued to the low-power DDR2-SDRAM. Program the NOP command in the Mode register (MPDDRC_MR). The application must configure the MODE field to 1 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The clocks which drive the Low-power DDR2-SDRAM devices are now enabled.
5. A pause of at least 100 ns must be observed before a signal toggle.
6. A NOP command is issued to the low-power DDR2-SDRAM. Program the NOP command in the MPDDRC_MR. The application must configure the MODE field to 1 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. CKE is now driven high.
7. A pause of at least 200 µs must be observed before issuing a Reset command.
8. A Reset command is issued to the low-power DDR2-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 63. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Reset command is now issued.
9. A pause of at least $t_{INIT5}$ must be observed before issuing any commands.
10. A Calibration command is issued to the low-power DDR2-SDRAM. Program the type of calibration in the Configuration register (MPDDRC_CR): configure the ZQ field to 3. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 10. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The ZQ Calibration command is now issued. Program the type of calibration in the MPDDRC_CR: configure the ZQ field to 2.
11. A Mode register Write command is issued to the low-power DDR2-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 1. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode register Write command is now issued.
12. A Mode register Write command is issued to the low-power DDR2-SDRAM. In the MPDDRC_MR, configure the MODE field to 7 and the MRS field to 2. The Mode register Write command cycle is issued to program parameters of the low-power DDR2-SDRAM device, in particular CAS latency. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode register Write command is now issued.
13. A Mode register Write command is issued to the low-power DDR2-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 3. The Mode register Write command cycle is issued to program parameters of the low-power DDR2-SDRAM device, in particular Drive Strength and Slew Rate. Read the MPDDRC_MR and add a memory barrier assembler instruction just after

the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode register Write command is now issued.

14. A Mode register Write command is issued to the low-power DDR2-SDRAM. In MPDDRC_MR configure the MODE field to 7 and the MRS field to 16. Mode register Write command cycle is issued to program parameters of the low-power DDR2-SDRAM device, in particular Partial Array Self Refresh (PASR). Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode register Write command is now issued.
15. In the DDR Configuration register (SFR_DDRCFG), the application must write a '1' to bits 17 and 16 to open the input buffers (refer to section "Special Function Registers (SFR)").
16. A NOP command is issued to the low-power DDR2-SDRAM. Program the NOP command in the Mode register (MPDDRC_MR). The application must configure the MODE field to 1 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command.
17. A Mode register Read command is issued to the low-power DDR2-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 5. The Mode register Read command cycle is used to read the LPDDR2 Manufacturer ID from the low-power DDR2-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode register Read command is now issued. The LPDDR2 Manufacturer ID is set in MPDDRC_MD. See MPDDRC Memory Device Register.
18. A Mode register Read command is issued to the low-power DDR2-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 6. The Mode register Read command cycle is used to read Revision ID1 from the low-power DDR2-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode register Read command is now issued. Revision ID1 is set in register MPDDRC_MD. See MPDDRC Memory Device Register.
19. A Mode register Read command is issued to the low-power DDR2-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 8. The Mode register Read command cycle is used to read the memory organization (I/O width, Density, Type) from the low-power DDR2-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode register Read command is now issued. Memory organization is set in register MPDDRC_MD. See MPDDRC Memory Device Register.
20. A Mode register Read command is issued to the low-power DDR2-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 0. The Mode register Read command cycle is used to read device information (RZQI, DAI) from the low-power DDR2-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode register Read command is now issued. Device information RZQI is set in register Timing Calibration (see MPDDRC Low-power DDR2 Low-power DDR3 and DDR3 Timing Calibration Register) and DAI is set in Mode register (see MPDDRC Mode Register).
21. A Normal Mode command is provided. Program the Normal mode in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command.

22. In the DDR configuration register (SFR_DDRCCFG), the application must write a '0' to bits 17 and 16 to close the input buffers. The buffers are then driven by the HMPDDRC controller.
23. Write the refresh rate into the COUNT field in the Refresh Timer register (MPDDRC_RTR). To compute the value, see MPDDRC Refresh Timer Register.

After initialization, the low-power DDR2-SDRAM devices are fully functional.

### 35.4.4 DDR3-SDRAM/DDR3L-SDRAM Initialization

The initialization sequence is generated by software. The DDR3-SDRAM devices are initialized by the following sequence:

1. Program the memory device type in the Memory Device register (MPDDRC_MD).
2. Program the shift sampling value in the Read Data Path register (MPDDRC_RD_DATA_PATH)
3. Program features of the DDR3-SDRAM device in the Configuration register (MPDDRC_CR) (number of columns, rows, banks, CAS latency and output driver impedance control) and in the Timing Parameter 0 register/Timing Parameter 1 register (MPDDRC_TPR0/1) (asynchronous timing - TRC, TRAS, etc.).
4. A NOP command is issued to the DDR3-SDRAM. Program the NOP command in the Mode register (MPDDRC_MR). The application must configure the MODE field to 1 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR3-SDRAM address to acknowledge this command. The clocks which drive the DDR3-SDRAM device are now enabled.
5. A pause of at least 500 µs must be observed before a signal toggle.
6. A NOP command is issued to the DDR3-SDRAM. Program the NOP command in the MPDDRC_MR. The application must configure the MODE field to 1 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR3-SDRAM address to acknowledge this command. CKE is now driven high.
7. An Extended Mode Register Set (EMRS2) cycle is issued to choose between commercial or high temperature operations. The application must configure the MODE field to 5 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR3-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[2] is set to 0, BA[1] is set to 1 and signal BA[0] is set to 0. For example: with a 16-bit, 1-Gbit, DDR3-SDRAM (14 rows, 10 columns, 8 banks), the DDR3-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x040000000; with a 32-bit, 1-Gbit, DDR3-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x08000000.
   **Note:** This address is given as an example only. The real address depends on the implementation in the product.
8. An Extended Mode Register Set (EMRS3) cycle is issued to set the Extended Mode register to 0. The application must configure the MODE field to 5 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR3-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[2] is set to 0, BA[1] is set to 1 and signal BA[0] is set to 1. For example: with a 16-bit, 1-Gbit, DDR3-SDRAM (14 rows, 10 columns, 8 banks), the DDR3-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x06000000; with a 32-bit, 1-Gbit, DDR3-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x0C000000.
9. An Extended Mode Register Set (EMRS1) cycle is issued to disable and to program ODS (output drive strength). The application must configure the MODE field to 5 in the MPDDRC_MR. Read the

MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR3-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[2:1] is set to 0 and signal BA[0] is set to 1. For example: with a 16-bit, 1-Gbit, DDR3-SDRAM (14 rows, 10 columns, 8 banks), the DDR3-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x02000000; with a 32-bit, 1-Gbit, DDR3-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x04000000.

10. Write a '1' to the DLL bit (enable DLL reset) in the Configuration register (MPDDRC_CR).
11. A Mode Register Set (MRS) cycle is issued to reset DLL. The application must configure the MODE field to 3 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR3-SDRAM to acknowledge this command. The write address must be chosen so that signals BA[2:0] are set to 0. For example, the SDRAM write access should be done at the address: BASE_ADDRESS_DDR.
12. A Calibration command (MRS) is issued to calibrate RTT and RON values for the Process Voltage Temperature (PVT). The application must configure the MODE field to 6 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR3-SDRAM to acknowledge this command. The write address must be chosen so that signals BA[2:0] are set to 0. For example, the SDRAM write access should be done at the address: BASE_ADDRESS_DDR.
13. A Normal Mode command is provided. Program the Normal mode in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR3-SDRAM address to acknowledge this command.
14. Write the refresh rate into the COUNT field in the Refresh Timer register (MPDDRC_RTR). To compute the value, see MPDDRC Refresh Timer Register.

After initialization, the DDR3-SDRAM devices are fully functional.

### 35.4.5 Low-power DDR3-SDRAM Initialization

The initialization sequence is generated by software. The low-power DDR3-SDRAM devices are initialized by the following sequence:

1. Program the memory device type in the Memory Device register (MPDDRC_MD).
2. Program the shift sampling value in the Read Data Path register (MPDDRC_RD_DATA_PATH).
3. Program features of the low-power DDR3-SDRAM device into and in the Configuration register (MPDDRC_CR) (number of columns, rows, banks, CAS latency and output drive strength) and in the Timing Parameter 0 register/Timing Parameter 1 register (MPDDRC_TPR0/1) (asynchronous timing: TRC, TRAS, etc.).
4. A NOP command is issued to the low-power DDR3-SDRAM. Program the NOP command in the Mode register (MPDDRC_MR). The application must configure the MODE field to 1 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The clocks which drive the low-power DDR3-SDRAM devices are now enabled.
5. A pause of at least 100 ns must be observed before a signal toggle.
6. A NOP command is issued to the low-power DDR3-SDRAM. Program the NOP command in the MPDDRC_MR. The application must configure the MODE field to 1 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. CKE is now driven high.
7. A pause of at least 200 µs must be observed before issuing a Reset command.

8. A Reset command is issued to the low-power DDR3-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 63. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Reset command is now issued.
9. A pause of at least $t_{INIT5}$ must be observed before issuing any commands.
10. A Calibration command is issued to the low-power DDR3-SDRAM. Program the type of calibration in the Configuration register (MPDDRC_CR): set the ZQ field to 3. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 10. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The ZQ Calibration command is now issued. Program the type of calibration in the MPDDRC_CR: set the ZQ field to 2.
11. A Mode register Write command is issued to the low-power DDR3-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 1. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode register Write command is now issued.
12. A Mode register Write command is issued to the low-power DDR3-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 2. The Mode register Write command cycle is issued to program parameters of the low-power DDR3-SDRAM device, in particular CAS Latency. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode register Write command is now issued.
13. A Mode register Write command is issued to the low-power DDR3-SDRAM. In the MPDDRC_MR, configure the MODE field 7 and the MRS field to 3. The Mode register Write command cycle is issued to program parameters of the low-power DDR3-SDRAM device, in particular Drive Strength and Slew Rate. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode register Write command is now issued.
14. A Mode register Write command is issued to the low-power DDR3-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 16. The Mode register Write command cycle is issued to program parameters of the low-power DDR3-SDRAM device, in particular Partial Array Self Refresh (PASR). Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode register Write command is now issued.
15. In the DDR Configuration register (SFR_DDRCFG), the application must write a ‘1’ to bits 17 and 16 to open the input buffers.
16. A NOP command is issued to the low-power DDR3-SDRAM. Program the NOP command in the Mode register (MPDDRC_MR). The application must configure the MODE field to 1 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command.
17. A Mode register Read command is issued to the low-power DDR3-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 5. The Mode register Read command cycle is used to read the LPDDR3 Manufacturer ID from the low-power DDR3-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode register Read command is now issued. The LPDDR3 Manufacturer ID is set in register MPDDRC_MD. See MPDDRC Memory Device Register.

18. A Mode register Read command is issued to the low-power DDR3-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 6. The Mode register Read command cycle is used to read the Revision ID1 from the low-power DDR3-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode register Read command is now issued. Revision ID1 is set in register MPDDRC_MD. See MPDDRC Memory Device Register.
19. A Mode register Read command is issued to the low-power DDR3-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 8. The Mode register Read command cycle is used to read memory organization (I/O width, Density, Type) from the low-power DDR3-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode register Read command is now issued. Memory organization is set in register MPDDRC_MD. See MPDDRC Memory Device Register.
20. A Mode register Read command is issued to the low-power DDR3-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 0. The Mode register Read command cycle is used to read the device information (RZQI, DAI) from the low-power DDR3-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode register Read command is now issued. Device information RZQI is set in register Timing Calibration (see MPDDRC Low-power DDR2 Low-power DDR3 and DDR3 Timing Calibration Register) and DAI is set in Mode register (see MPDDRC Mode Register).
21. A Normal Mode command is provided. Program the Normal mode in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command.
22. In the DDR Configuration register (SFR_DDRCCFG), the application must write a ‘0’ to bits 17 and 16 to close the input buffers. The buffers are then driven by the HMPDDRC controller.
23. Write the refresh rate into the COUNT field in the Refresh Timer register (MPDDRC_RTR). To compute the value, see MPDDRC Refresh Timer Register.

After initialization, the low-power DDR3-SDRAM devices are fully functional.

## 35.5 Functional Description

### 35.5.1 DDR-SDRAM Controller Write Cycle

The MPDDRC provides burst access or single access in Normal mode (MPDDRC_MR.MODE = 0). Whatever the access type, the MPDDRC keeps track of the active row in each bank, thus maximizing performance.

The DDR-SDRAM device is programmed with a burst length (bl) equal to 8. This determines the length of a sequential data input by the write command that is set to 8. The latency from write command to data input depends on the memory type, as shown in the following table.

**Table 35-1. CAS Write Latency**

| Memory Devices | CAS Write Latency (CWL) |
|---|---|
| Low-power DDR1-SDRAM | 1 |
| Low-power DDR2-SDRAM | 1 |

| .....continued | |
|---|---|
| **Memory Devices** | **CAS Write Latency (CWL)** |
| DDR2-SDRAM | 2 |
| Low-power DDR3-SDRAM | 1/3 |
| DDR3-SDRAM (DLL Off) | 6 |

**Note:** In the case of low-power DDR3-SDRAM, the CAS Write Latency (CWL) of 1 is optional. The MPDDRC supports this feature. Refer to the low-power DDR3-SDRAM datasheet for details.

To initiate a single access, the MPDDRC checks if the page access is already open. If row/bank addresses match with the previous row/bank addresses, the controller generates a write command. If the bank addresses are not identical or if bank addresses are identical but the row addresses are not identical, the controller generates a precharge command, activates the new row and initiates a write command. To comply with DDR-SDRAM timing parameters, additional clock cycles are inserted between precharge/active ($t_{RP}$) commands and active/write ($t_{RCD}$) commands. As the burst length is set to 8, in case of single access, it has to stop the burst, otherwise seven invalid values may be written. In case of the DDR-SDRAM device, the burst stop command is not supported for the burst write operation. Thus, in order to interrupt the write operation, the DM (data mask) input signal must be set to 1 to mask invalid data (see Figures Single Write Access, Row Closed, DDR-SDRAM Devices and Burst Write Access, Row Closed, DDR-SDRAM Devices), and DQS must continue to toggle.

To initiate a burst access, the MPDDRC uses the transfer type signal provided by the master requesting the access. If the next access is a sequential write access, writing to the DDR-SDRAM device is carried out. If the next access is a write non-sequential access, then an automatic access break is inserted, the MPDDRC generates a precharge command, activates the new row and initiates a write command. To comply with DDR-SDRAM timing parameters, additional clock cycles are inserted between precharge/active ($t_{RP}$) commands and active/write ($t_{RCD}$) commands.

For the definition of timing parameters, see MPDDRC Timing Parameter 0 Register.

Write accesses to the DDR-SDRAM device are burst oriented and the burst length is programmed to 8. It determines the maximum number of column locations that can be accessed for a given write command. When the write command is issued, eight columns are selected. All accesses for that burst take place within these eight columns, thus the burst wraps within these eight columns if a boundary is reached. These eight columns are selected by addr[13:3]. addr[2:0] is used to select the starting location within the block.

In case of incrementing burst (INCR/INCR4/INCR8/INCR16), the addresses can cross the 16-byte boundary of the DDR-SDRAM device. For example, when a transfer (INCR4) starts at address 0x0C, the next access is 0x10, but since the burst length is programmed to 8, the next access is at 0x00. Since the boundary is reached, the burst is wrapped. The MPDDRC takes this feature of the DDR-SDRAM device into account. In case of a transfer starting at address 0x04/0x08/0x0C or starting at address 0x10/0x14/0x18/0x1C, two write commands are issued to avoid wrapping when the boundary is reached. The last write command is subject to DM input logic level. If DM is registered high, the corresponding data input is ignored and the write access is not done. This avoids additional writing.

**Figure 35-2. Single Write Access, Row Closed, DDR-SDRAM Devices**

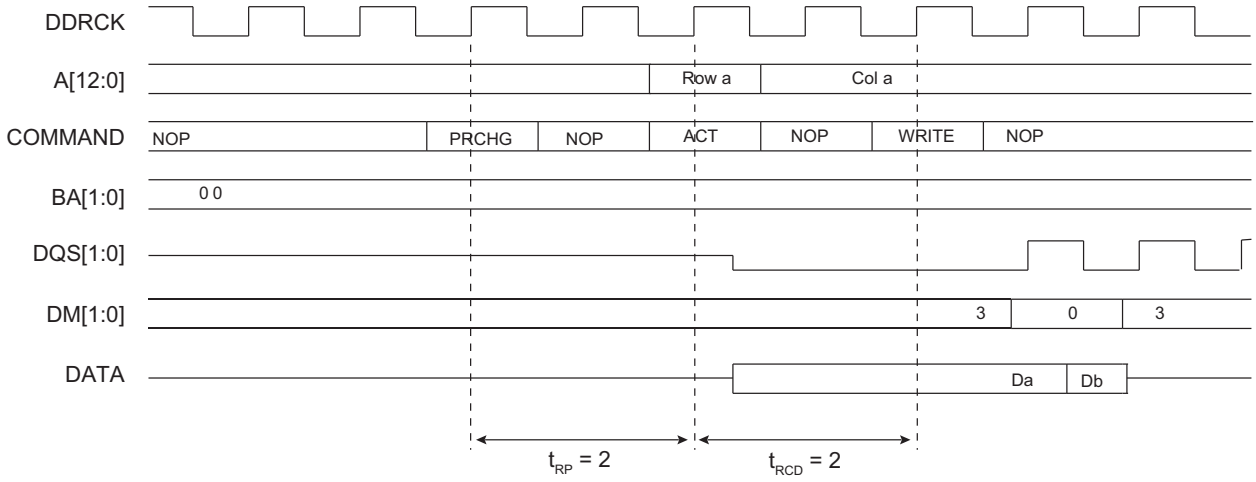



**Figure 35-3. Single Write Access, Row Closed, DDR2-SDRAM Devices**

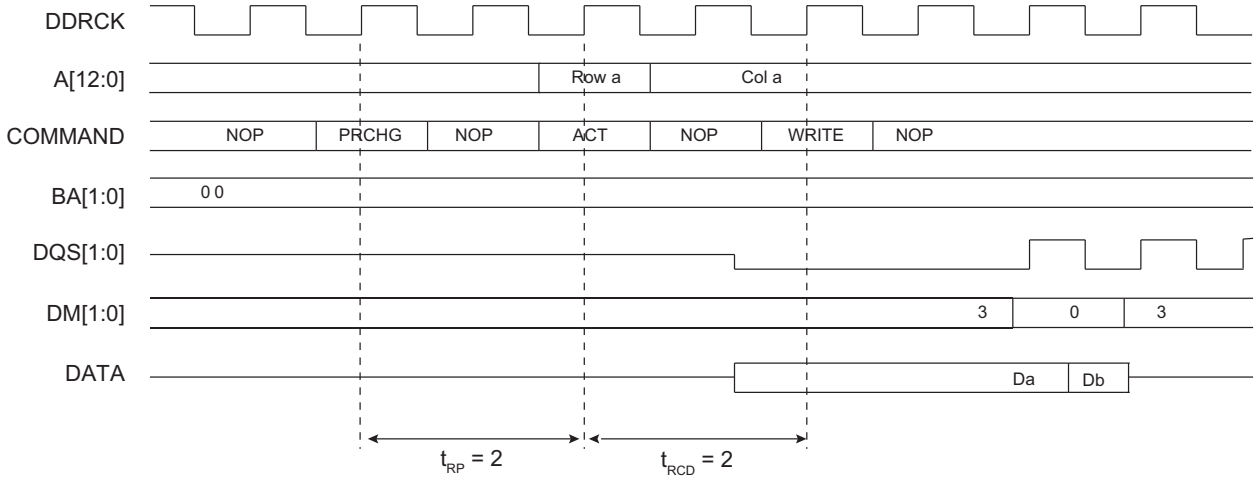



**Figure 35-4. Burst Write Access, Row Closed, DDR-SDRAM Devices**

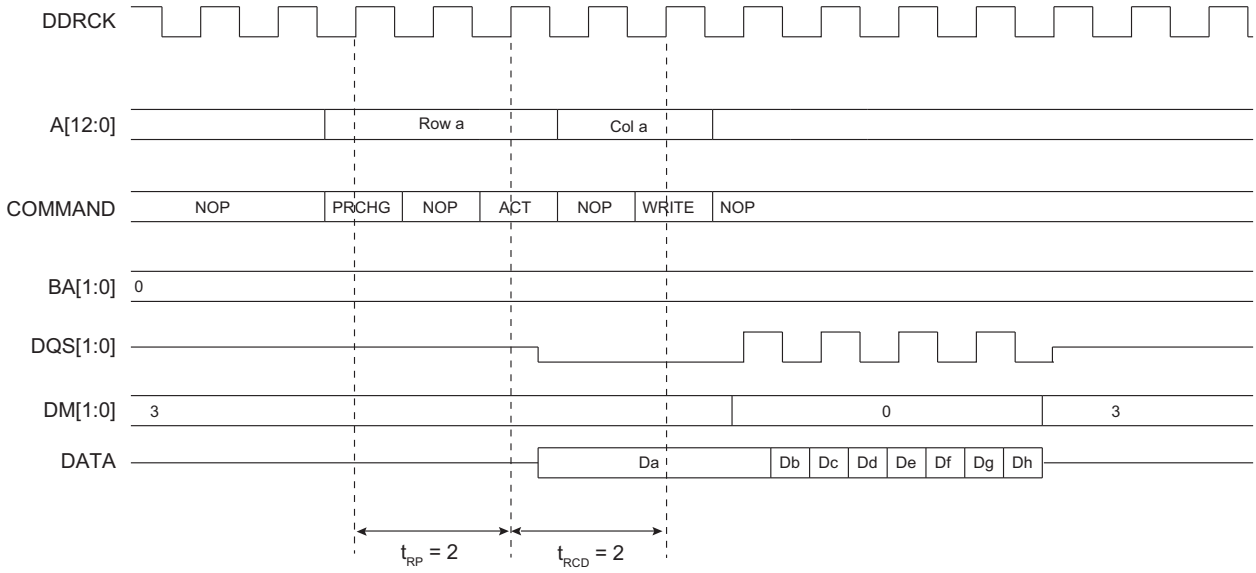
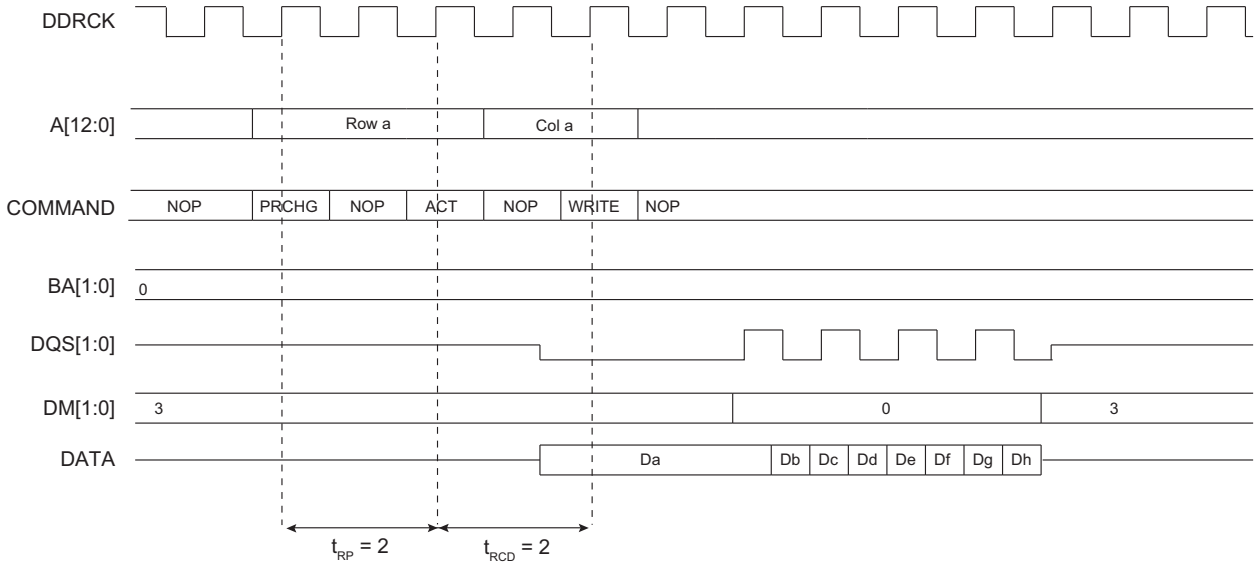
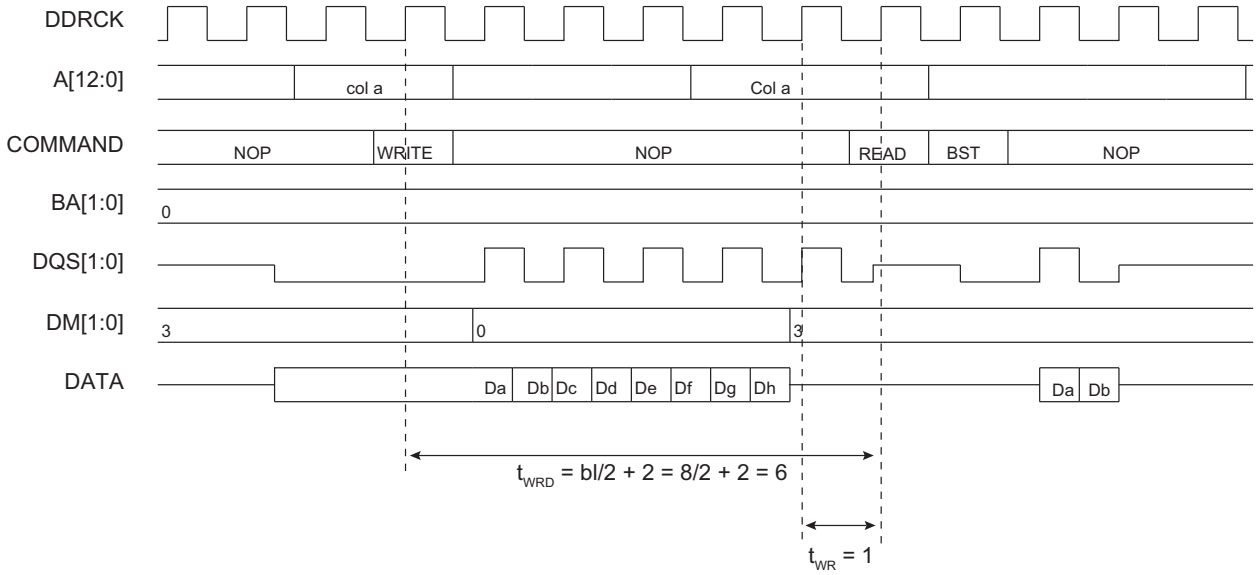


**Figure 35-5. Burst Write Access, Row Closed, DDR2-SDRAM Devices**




A write command can be followed by a read command. To avoid breaking the current write burst, $t_{WTR}$/ $t_{WRD}$ (bl/2 + 2 = 6 cycles) should be met. See the figure below.

**Figure 35-6. Write Command Followed by a Read Command without Burst Write Interrupt, DDR-SDRAM Devices**




In case of a single write access, write operation should be interrupted by a read access but DM must be input 1 cycle prior to the read command to avoid writing invalid data. See the figure below.

**Figure 35-7. Single Write Access Followed by a Read Access, DDR-SDRAM Devices**

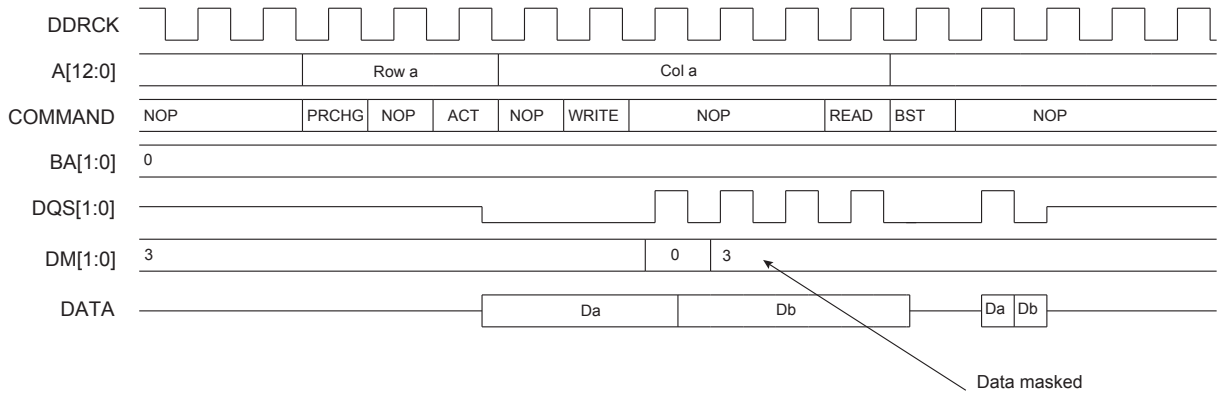
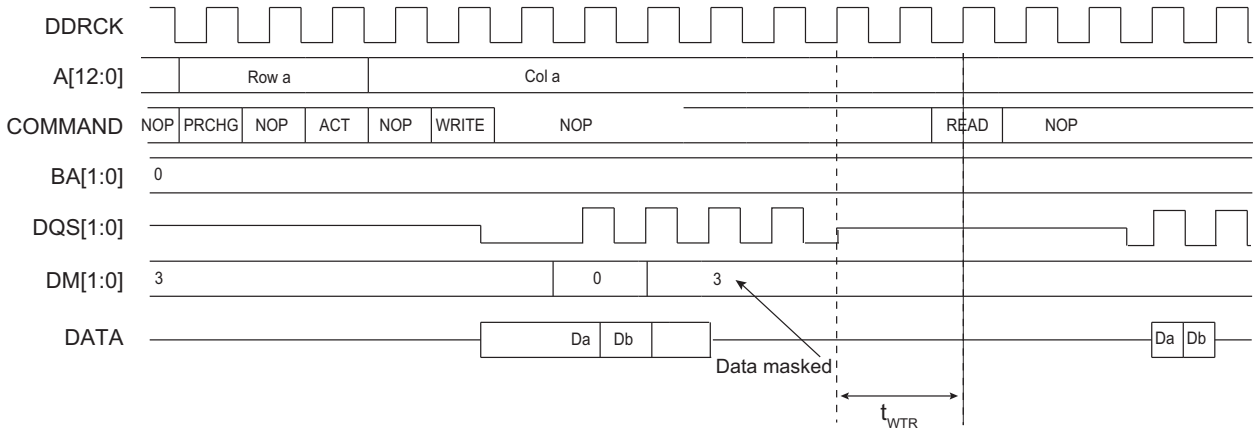



**Figure 35-8. Single Write Access Followed by a Read Access, DDR2-SDRAM Devices**




### 35.5.2 DDR-SDRAM Controller Read Cycle

The MPDDRC provides burst access or single access in Normal mode (MPDDRC_MR.MODE = 0). Whatever the access type, the MPDDRC keeps track of the active row in each bank, thus maximizing performance of the MPDDRC.

The DDR-SDRAM devices are programmed with a burst length equal to 8 which determines the length of a sequential data output by the read command that is set to 8. The latency from read command to data output depends on the memory type, as shown in the following table. This value is programmed during the initialization phase (see Product Dependencies, Initialization Sequence).

**Table 35-2. CAS Read Latency**

| Memory Devices | CAS Read Latency |
|---|---|
| Low-power DDR1-SDRAM | 2/3 |
| Low-power DDR2-SDRAM | 3 |
| DDR2-SDRAM | 3 |
| Low-power DDR3-SDRAM | 3/6 |
| DDR3-SDRAM (DLL Off) | 5/6 |

**Note:** In the case of low-power DDR3-SDRAM, the CAS Read Latency (CRL) of 3 is optional. The MPDDRC supports this feature. Refer to the low-power DDR3-SDRAM datasheet for details.

To initiate a single access, the MPDDRC checks if the page access is already open. If row/bank addresses match with the previous row/bank addresses, the controller generates a read command. If the bank addresses are not identical or if bank addresses are identical but the row addresses are not identical, the controller generates a precharge command, activates the new row and initiates a read command. To comply with DDR-SDRAM timing parameters, additional clock cycles are inserted between precharge/active ($t_{RP}$) commands and active/read ($t_{RCD}$) commands. After a read command, additional wait states are generated to comply with CAS latency. The MPDDRC supports a CAS latency delay of 2, 3, 5 or 6 clock cycles. As the burst length is set to 8, in case of a single access or a burst access inferior to 8 data requests, it has to stop the burst, otherwise an additional seven or X values could be read. The Burst Stop command (BST) is used to stop output during a burst read. If the DDR2-SDRAM Burst Stop command is not supported by the JEDEC standard, in a single read access, an additional seven unwanted data will be read.

To initiate a burst access, the MPDDRC checks the transfer type signal. If the next accesses are sequential read accesses, reading to the SDRAM device is carried out. If the next access is a read non-sequential access, then an automatic page break can be inserted. If the bank addresses are not identical or if bank addresses are identical but the row addresses are not identical, the controller generates a precharge command, activates the new row and initiates a read command. If page access is already open, a read command is generated.

To comply with DDR-SDRAM timing parameters, additional clock cycles are inserted between precharge/active ($t_{RP}$) commands and active/read ($t_{RCD}$) commands. The MPDDRC supports a CAS latency delay of 2, 3, 5 or 6 clock cycles. During this delay, the controller uses internal signals to anticipate the next access and improve the performance of the controller. Depending on the latency, the MPDDRC anticipates 2, 3, 5 or 6 read accesses. In case of burst of specified length, accesses are not anticipated, but if the burst is broken (border, Busy mode, etc.), the next access is treated as an incrementing burst of unspecified length, and depending on the latency, the MPDDRC anticipates 2, 3, 5 or 6 read accesses.

For the definition of timing parameters, see MPDDRC Configuration Register.

Read accesses to the DDR-SDRAM are burst oriented and the burst length is programmed to 8. The burst length determines the maximum number of column locations that can be accessed for a given read command. When the read command is issued, eight columns are selected. All accesses for that burst take place within these eight columns, meaning that the burst wraps within these eight columns if the boundary is reached. These eight columns are selected by addr[13:3]; addr[2:0] is used to select the starting location within the block.

In case of incrementing burst (INCR/INCR4/INCR8/INCR16), the addresses can cross the 16-byte boundary of the DDR-SDRAM device. For example, when a transfer (INCR4) starts at address 0x0C, the next access is 0x10, but since the burst length is programmed to 8, the next access is 0x00. Since the boundary is reached, the burst wraps. The MPDDRC takes into account this feature of the SDRAM device. In case of the DDR-SDRAM device, transfers start at address 0x04/0x08/0x0C. Two read commands are issued to avoid wrapping when the boundary is reached. The last read command may generate additional reading (1 read cmd = 4 DDR words).

To avoid additional reading, it is possible to use the burst stop command to truncate the read burst and to decrease power consumption. The DDR2-SDRAM devices do not support the burst stop command.

**Figure 35-9. Single Read Access, Row Closed, Latency = 2, DDR-SDRAM Devices**

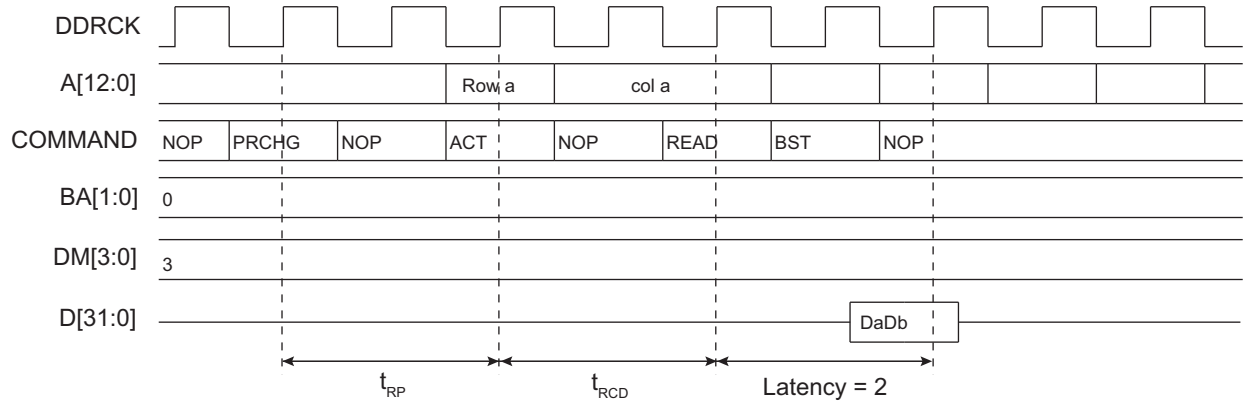



**Figure 35-10. Single Read Access, Row Closed, Latency = 3, DDR2-SDRAM Devices**

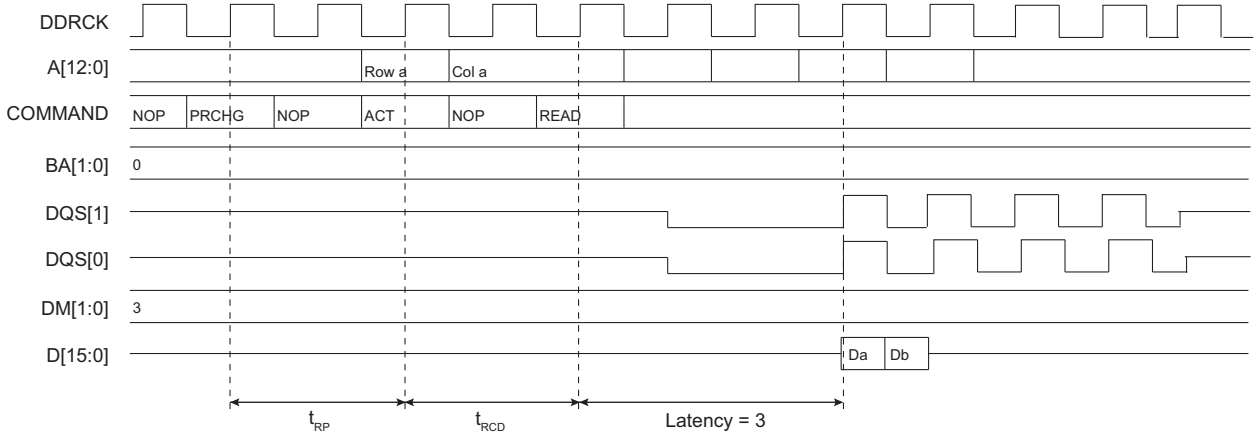



**Figure 35-11. Burst Read Access, Latency = 2, DDR-SDRAM Devices**

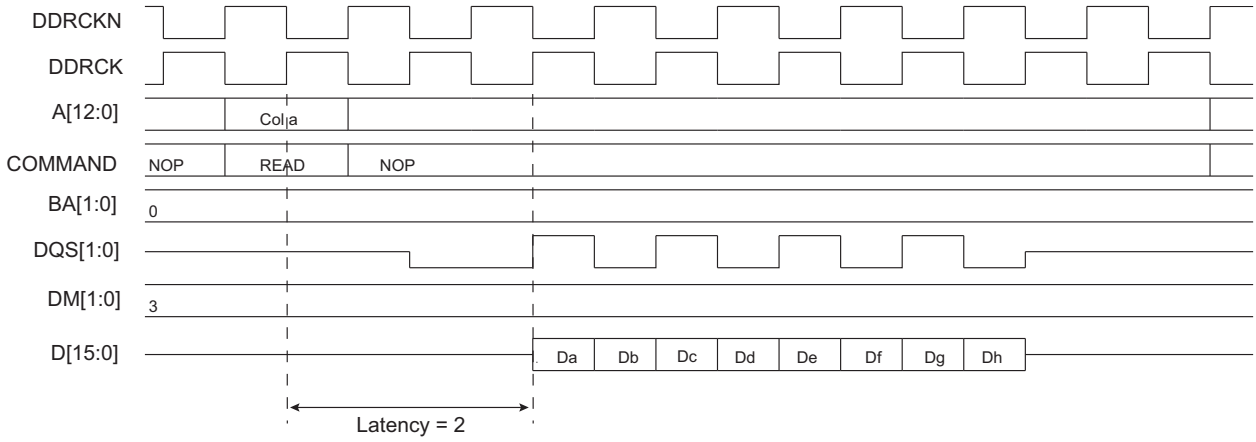
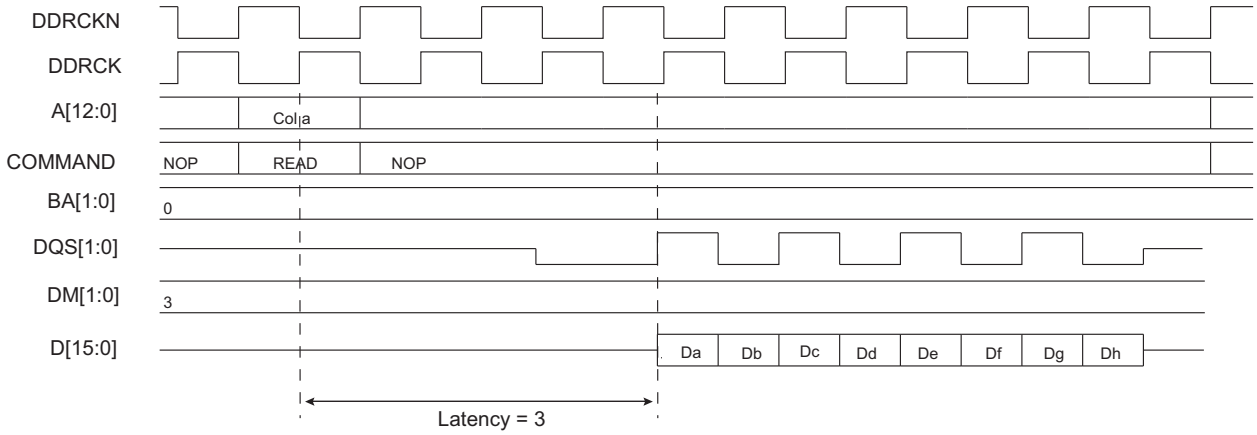


**Figure 35-12. Burst Read Access, Latency = 3, DDR2-SDRAM Devices**




### 35.5.3 Refresh (Autorefresh Command)

#### 35.5.3.1 All Banks Autorefresh

The All Banks Autorefresh command performs a refresh operation on all banks. An autorefresh command is used to refresh the external device. Refresh addresses are generated internally by the DDR-SDRAM device and incremented after each autorefresh automatically. The MPDDRC generates these autorefresh commands periodically. A timer is loaded in the MPDDRC_RTR with the value that indicates the number of clock cycles between refresh cycles (see MPDDRC Refresh Timer Register). When the MPDDRC initiates a refresh of the DDR-SDRAM device, internal memory accesses are not delayed. However, if the CPU tries to access the DDR-SDRAM device, the slave indicates that the device is busy. A refresh request does not interrupt a burst transfer in progress. This feature is activated by setting Per-bank Refresh bit (REF_PB) to 0 in the MPDDRC_RTR (see MPDDRC Refresh Timer Register).

#### 35.5.3.2 Per-bank Autorefresh

The low-power DDR2-SDRAM and low-power DDR3-SDRAM embeds a new Per-bank Refresh command which performs a refresh operation on the bank scheduled by the bank counter in the memory device. The Per-bank Refresh command is executed in a fixed sequence order of round-robin type: “0-1-2-3-4-5-6-7-0-1-...”. The bank counter is automatically cleared upon issuing a RESET command or when exiting from Self-refresh mode, in order to ensure the synchronism between SDRAM memory device and the MPDDRC. The bank addressing for the Per-bank Refresh count is the same as established in the Single-bank Precharge command. This feature is activated by setting the Per-bank Refresh bit (REF_PB) to 1 in the MPDDRC_RTR (see MPDDRC Refresh Timer Register). This feature masks the latency due to the refresh procedure. The target bank is inaccessible during the Per-bank Refresh cycle period ($t_{RFCpb}$), however other banks within the device are accessible and may be addressed during the “Per-bank Refresh” cycle. During the REFpb operation, any bank other than the one being refreshed can be maintained in active state or accessed by a read or a write command. When the “Per-bank Refresh” cycle is completed, the affected bank will be in idle state.

#### 35.5.3.3 Adjust Autorefresh Rate

The low-power DDR2-SDRAM and low-power DDR3-SDRAM embeds an internal register, Mode register 19 (Refresh mode). The content of this register allows to adjust the interval of autorefresh operations according to temperature variation. This feature is activated by setting the Adjust Refresh bit [ADJ_REF] to 1 in the MPDDRC_RTR (see MPDDRC Refresh Timer Register). When this feature is enabled, a Mode Register Read (MRR) command is performed every 16 × $t_{REFI}$ (average time between REFRESH commands). Depending on the read value, the autorefresh interval will be modified. In case of high temperature, the interval is reduced and in case of low temperature, the interval is increased.

### 35.5.4 Power Management

#### 35.5.4.1 Self-refresh Mode

This mode is activated by configuring the Low-power Command bit (LPCB) to 1 in the MPDDRC Low-Power Register (MPDDRC_LPR).

Self-refresh mode is used in Powerdown mode, i.e., when no access to the DDR-SDRAM device is possible. In this case, power consumption is very low. In Self-refresh mode, the DDR-SDRAM device retains data without external clocking and provides its own internal clocking, thus performing its own autorefresh cycles. During the self-refresh period, CKE is driven low. As soon as the DDR-SDRAM device is selected, the MPDDRC provides a sequence of commands and exits Self-refresh mode.

The MPDDRC reenables Self-refresh mode as soon as the DDR-SDRAM device is not selected. It is possible to define when Self-refresh mode is to be enabled by configuring the TIMEOUT field in the MPDDRC_LPR:

0: Self-refresh mode is enabled as soon as the DDR-SDRAM device is not selected.

1: Self-refresh mode is enabled 64 clock cycles after completion of the last access.

2: Self-refresh mode is enabled 128 clock cycles after completion of the last access.

This controller also interfaces the low-power DDR-SDRAM. To optimize power consumption, the Low Power DDR SDRAM provides programmable self-refresh options comprised of Partial Array Self Refresh (full, half, quarter and 1/8 and 1/16 array).

Disabled banks are not refreshed in Self-refresh mode. This feature permits to reduce the self-refresh current. In case of low-power DDR1-SDRAM, the Extended Mode register controls this feature. It includes Temperature Compensated Self-refresh (TSCR) and Partial Array Self-refresh (PASR) parameters and the drive strength (DS) (see MPDDRC Low-Power Register). In case of low-power DDR2-SDRAM and low-power DDR3-SDRAM, the Mode Registers 16 and 17 control this feature, including PASR Bank Mask (BK_MASK) and PASR Segment Mask (SEG_MASK) parameters and drives strength (DS) (see MPDDRC Low-power DDR2 Low-power DDR3 Low-power Register). These parameters are set during the initialization phase. After initialization, as soon as the PASR/DS/TCSR fields or BK_MASK/SEG_MASK/DS are modified, the memory device Extended Mode register or Mode registers 3/16/17 are automatically accessed. Thus if MPDDRC does not share an external bus with another controller, PASR/DS/TCSR and BK_MASK/SEG_MASK/DS bits are updated before entering Self-refresh mode or during a refresh command. If MPDDRC does share an external bus with another controller, PASR/DS/TCSR and BK_MASK/SEG_MASK/DS bits are also updated during a pending read or write access. This type of update depends on the UPD_MR bit (see MPDDRC Low-Power Register).

The low-power DDR1-SDRAM must remain in Self-refresh mode during the minimum of TRFC periods (see MPDDRC Timing Parameter 1 Register), and may remain in Self-refresh mode for an indefinite period.

The DDR2-SDRAM must remain in Self-refresh mode during the minimum of $t_{CKE}$ periods (see the memory device datasheet), and may remain in Self-refresh mode for an indefinite period.

The low-power DDR2-SDRAM and low-power DDR3-SDRAM must remain in Self-refresh mode for the minimum of $t_{CKESR}$ periods (see the memory device datasheet) and may remain in Self-refresh mode for an indefinite period.

The DDR3-SDRAM must remain in Self-refresh mode for the minimum of $t_{CKESR}$ periods (see the memory device datasheet) and may remain in Self-refresh mode for an indefinite period.

**Figure 35-13. Self-refresh Mode Entry, TIMEOUT = 0**

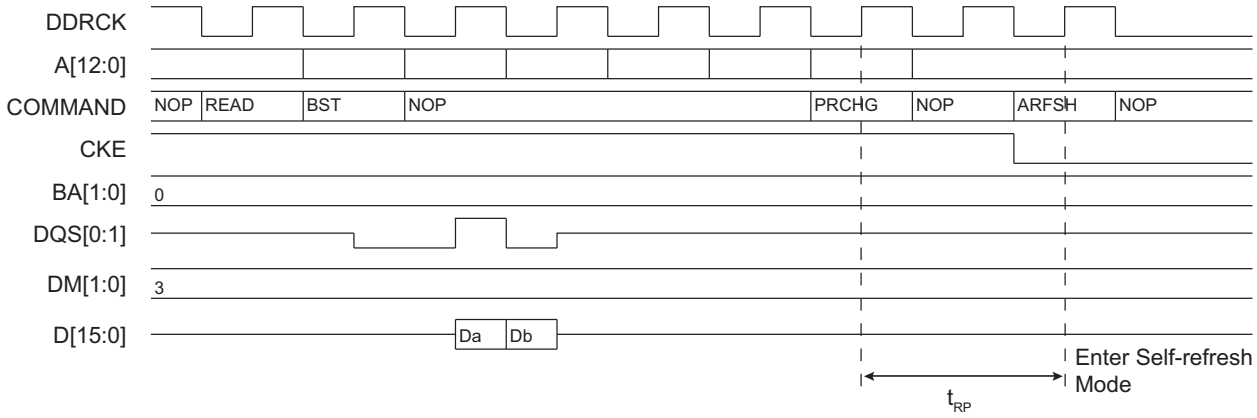



**Figure 35-14. Self-refresh Mode Entry, TIMEOUT = 1 or 2**

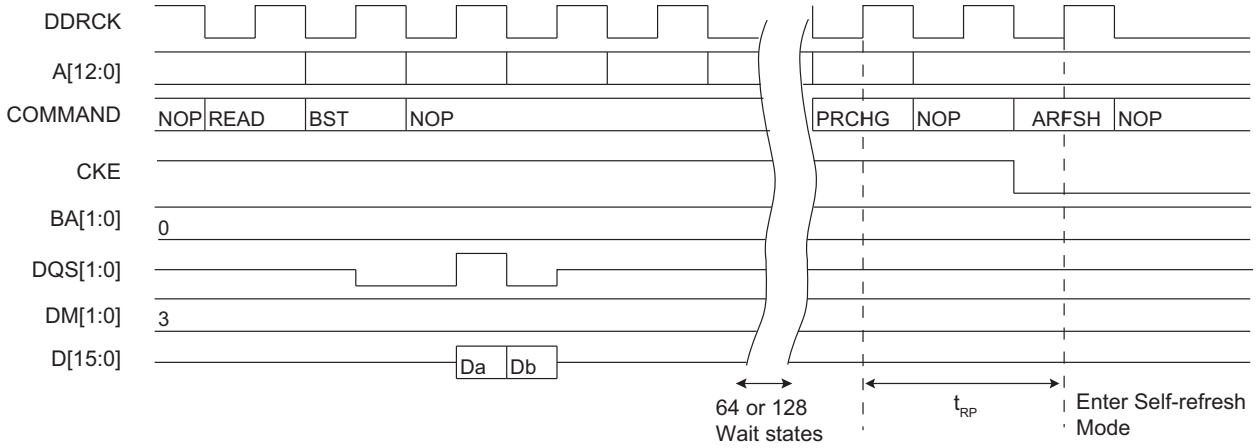



**Figure 35-15. Self-refresh Mode Exit**

### 35.5.4.2 Powerdown Mode

This mode is activated by configuring the Low-power Command bit (LPCB) to 2 in the MPDDRC Low-Power Register (MPDDRC_LPR).

Powerdown mode is used when no access to the DDR-SDRAM device is possible. In this mode, power consumption is greater than in Self-refresh mode. This state is similar to Normal mode (no Low-power

mode/no Self-refresh mode), but the CKE pin is low and the input and output buffers are deactivated as soon the DDR-SDRAM device is no longer accessible. In contrast to Self-refresh mode, the DDR-SDRAM device cannot remain in Low-power mode longer than one refresh period (64 ms/32 ms). As no autorefresh operations are performed in this mode, the MPDDRC carries out the refresh operation. For the low-power DDR-SDRAM devices, a NOP command must be generated for a minimum period defined in the TXP field of the Timing Parameter 1 register (MPDDRC_TPR1). For DDR-SDRAM devices, a NOP command must be generated for a minimum period defined in the TXP field of MPDDRC_TPR1 (see MPDDRC Timing Parameter 1 Register) and in the TXARD and TXARDS fields of MPDDRC_TPR2 (see MPDDRC Timing Parameter 2 Register) for DDR2_SDRAM devices. In addition, low-power DDR-SDRAM and DDR-SDRAM must remain in Powerdown mode for a minimum period corresponding to $t_{CKE}$, $t_{PD}$, etc. (refer to the memory device datasheet).

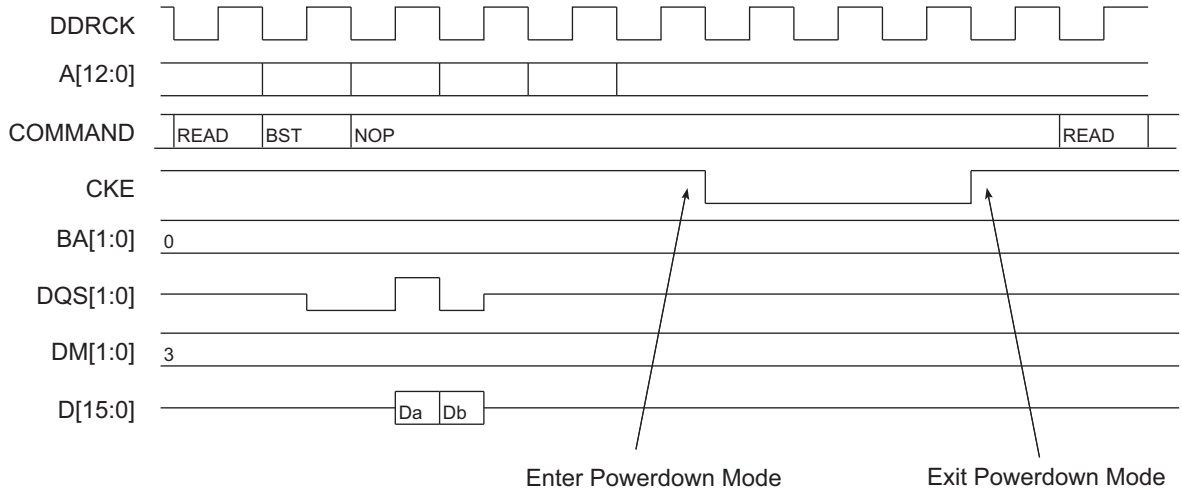
The exit procedure is faster than in Self-refresh mode. See the following figure. The MPDDRC returns to Powerdown mode as soon as the DDR-SDRAM device is not selected. It is possible to define when Powerdown mode is enabled by configuring the TIMEOUT field in the MPDDRC_LPR:

0: Powerdown mode is enabled as soon as the DDR-SDRAM device is not selected.

1: Powerdown mode is enabled 64 clock cycles after completion of the last access.

2: Powerdown mode is enabled 128 clock cycles after completion of the last access.

**Figure 35-16. Powerdown Entry/Exit, TIMEOUT = 0**



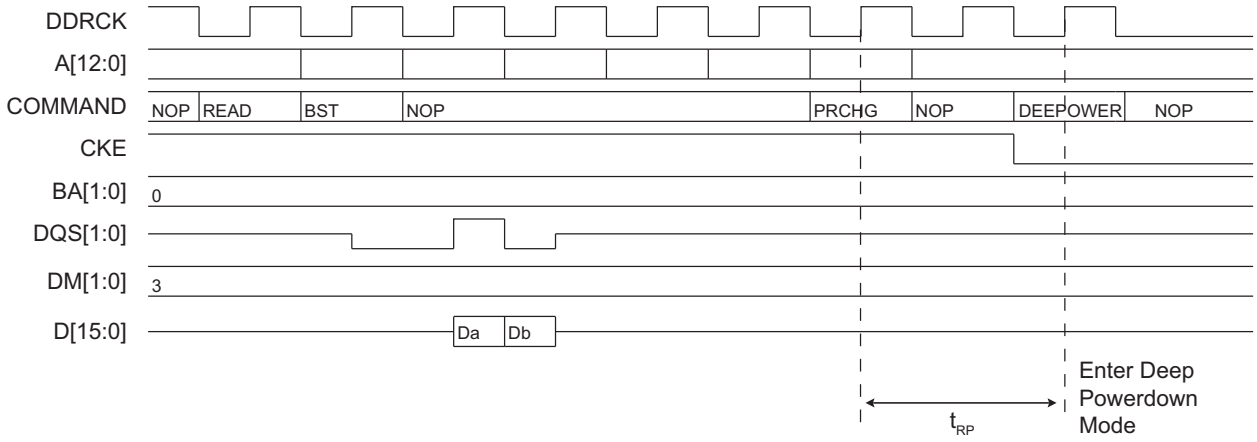

### 35.5.4.3 Deep Powerdown Mode

The Deep Powerdown mode is a feature of low-power DDR-SDRAM. When this mode is activated, all internal voltage generators inside the device are stopped and all data is lost.

Deep Powerdown mode is activated by configuring the Low-power Command bit (LPCB) to 3 in the MPDDRC Low-Power Register (MPDDRC_LPR). When this mode is enabled, the MPDDRC leaves Normal mode (MPDDRC_MR.MODE = 0) and the controller is frozen. The clock can be stopped during Deep Powerdown mode by setting the CLK_FR field to 1.

Before enabling this mode, the user must make sure there is no access in progress. To exit Deep Powerdown mode, the Low-power Command bit (LPCB) and Clock Frozen bit (CLK_FR) must be 0 and the initialization sequence must be generated by software. See Low-power DDR1-SDRAM Initialization or Low-power DDR2-SDRAM Initialization or Low-power DDR3-SDRAM Initialization.

**Figure 35-17. Deep Powerdown Mode Entry**




#### 35.5.4.4 Change Frequency During Self-Refresh Mode with Low-power DDR-SDRAM Devices and DDR3-SDRAM

To change frequency, Self-refresh mode must be activated. This is done by configuring the Low-power Command bit (LPCB) to 1 and writing a '1' to the Change Frequency Command bit (CHG_FR) in the Low-power register (MPDDRC_LPR).

Once the low-power DDR-SDRAM is in Self-refresh mode, the user must make sure there is no access in progress. Then, the user can change the clock frequency. The device input clock frequency changes only within minimum and maximum operating frequencies as specified by the low-power DDR-SDRAM and DDR3-SDRAM providers. Once the input clock frequency is changed, new stable clocks must be provided to the device before exiting from Self-refresh mode.

To exit from Self-refresh mode, the DDR-SDRAM device must be selected. The MPDDRC provides a sequence of commands and exits Self-refresh mode.

During a change frequency procedure, the Change Frequency Command bit (CHG_FR) is set to 0 automatically.

The Enable Read Measure feature is not supported during a change frequency procedure (see "ENRDM: Enable Read Measure").

It is not possible to change the frequency with DDR2-SDRAM devices.

#### 35.5.4.5 Reset Mode

The Reset mode is a feature of DDR2-SDRAM. This mode is activated by configuring the Low-power Command bit (LPCB) to 3 and writing a '1' to the Clock Frozen Command bit (CLK_FR) in the Low-power register (MPDDRC_LPR).

When this mode is enabled, the MPDDRC leaves Normal mode (MPDDRC_MR.MODE = 0) and the controller is frozen. Before enabling this mode, the user must make sure there is no access in progress.

To exit Reset mode, the Low-power Command bit (LPCB) must be configured to 0, the Clock Frozen Command bit (CLK_FR) must be written to '0' and the initialization sequence must be generated by software (see DDR2-SDRAM Initialization).

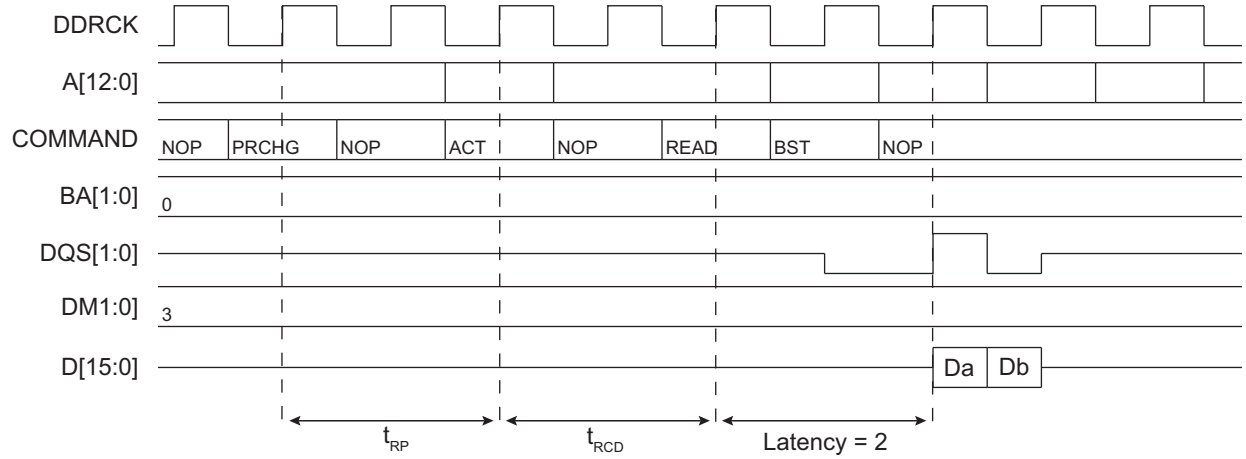
### 35.5.5 Multiport Functionality

The DDR-SDRAM protocol imposes a check of timings prior to performing a read or a write access, thus decreasing system performance. An access to DDR-SDRAM is performed if banks and rows are open (or active). To activate a row in a particular bank, the last open row must be deactivated and a new row must be open. Two DDR-SDRAM commands must be performed to open a bank: Precharge command and

Activate command with respect to $T_{RP}$ timing. Before performing a read or write command, $T_{RCD}$ timing must be checked.

This operation generates a significant bandwidth loss (see the following figure).

**Figure 35-18. $T_{RP}$ and $T_{RCD}$ Timings**

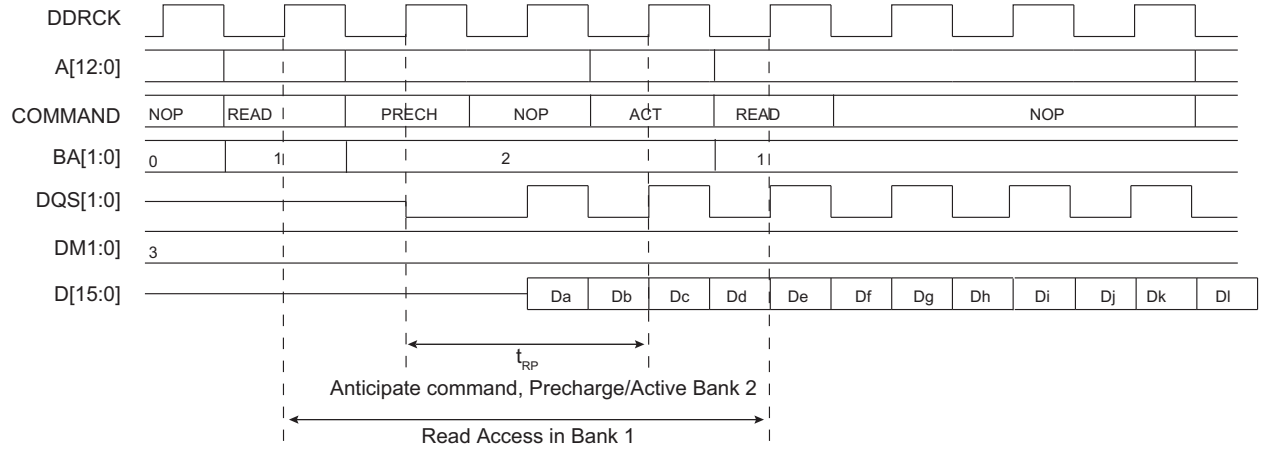



4 cycles before performing a read command

The multiport controller is designed to mask these timings and thus improve the system bandwidth.

The MPDDRC is a multiport controller whereby eight masters can simultaneously reach the controller. This feature improves the bandwidth of the system because it can detect eight requests on the AHB slave inputs and thus anticipate the commands that follow, Precharge command and Activate command in bank X during the current access in bank Y. This masks $t_{RP}$ and $t_{RCD}$ timings (see the following figure). In the best case, all accesses are done as if the banks and rows were already open. The best condition is met when the eight masters work in different banks. In the case of eight simultaneous read accesses, when the four or eight banks and associated rows are open, the controller reads with a continuous flow and masks the CAS latency for each access. To allow a continuous flow, the read command must be set at 2, 3, 5 or 6 cycles (CAS latency) before the end of the current access. The arbitration scheme must be changed since the round-robin arbitration cannot be respected. If the controller anticipates a read access, and thus a master with a high priority arises before the end of the current access, then this master will not be serviced.

**Figure 35-19. Anticipate Precharge/Activate Command in Bank 2 during Read Access in Bank 1**



MPDDRC is a multiport controller that embeds three arbitration mechanisms based on round-robin arbitration which allows to share the external device between different masters when two or more masters try to access the DDR-SDRAM device at the same time.

The three arbitration types are round-robin arbitration and two weighted round-robin arbitrations. For weighted round-robin arbitrations, the priority can be given either depending on the number of requests or words per port, or depending on the required bandwidth per port. The type of arbitration can be chosen by setting the ARB field in the Configuration Arbiter register (MPDDRC_CONF_ARBITER) (see MPDDRC Configuration Arbiter Register).

#### 35.5.5.1 Round-robin Arbitration

Round-robin arbitration is used when the ARB field is set to 0 (see MPDDRC Configuration Arbiter Register). This algorithm dispatches the requests from different masters to the DDR-SDRAM device in a round-robin manner. If two or more master requests arise at the same time, the master with the lowest number is serviced first, then the others are serviced in a round-robin manner.

To avoid burst breaking and to provide the maximum throughput for the DDR-SDRAM device, arbitration must only take place during the following cycles:

1. Idle cycles: when no master is connected to the DDR-SDRAM device.
2. Single cycles: when a slave is currently doing a single access.
3. End of Burst cycles: when the current cycle is the last cycle of a burst transfer:
   - For bursts of defined length, predicted end of burst matches the size of the transfer.
   - For bursts of undefined length, predicted end of burst is generated at the end of each four-beat boundary inside the INCR transfer.
4. Anticipated Access: when an anticipated read access is done while the current access is not complete, the arbitration scheme can be changed if the anticipated access is not the next access serviced by the arbitration scheme.

#### 35.5.5.2 Request-word Weighted Round-robin Arbitration

In request-word weighted round-robin arbitration, the weight is the number of requests or the number of words per port.

This arbitration scheme is enabled by configuring the ARB field to 1 (see MPDDRC Configuration Arbiter Register). This algorithm grants a port for X(1) consecutive first transfer (htrans = NON SEQUENTIAL) of a burst or X single transfer, or for X word transfers. It is possible to choose between an arbitration scheme by request or by word per port by setting the RQ_WD_Px field (see MPDDRC Configuration Arbiter Register).

Note: 1. X is an integer value provided by some master modules to the arbiter.

It is also possible for the user to provide the number of requests or words (by overwriting the information provided by a master) on master basis by configuring the MA_PR_Px field. Depending on the application, it is possible to reduce or increase the number of these requests or words by configuring the NRD_NWD_BDW_Px fields (see MPDDRC Configuration Arbiter Register).

The TIMEOUT_Px field defines the delay between two accesses on the same port in number of cycles before rearbitrating the access to another port. This field allows to avoid a timeout on the system because some masters have the particularity to add idle cycles between two consecutive accesses (see MPDDRC Configuration Arbiter Register).

This algorithm dispatches the requests from different masters to the DDR-SDRAM device in a round-robin manner. If two or more master requests arise at the same time, the master with the lowest number is

serviced first, then the others are serviced in a round-robin manner when the number of requests or words is reached or when the timeout value is reached.

To avoid burst breaking and to provide the maximum throughput for the DDR-SDRAM device, arbitration must only take place during the following cycles:

1. Timeout is reached: the delay between two accesses is equal to TIMEOUT_Px.
2. Number of requests or words is reached: when the current cycle is the last cycle of a transfer.

#### 35.5.5.3 Bandwidth Weighted Round-robin Arbitration

In bandwidth weighted round-robin arbitration, a minimum bandwidth is guaranteed per port.

This arbitration scheme is enabled when the ARB field is set to 2 (see MPDDRC Configuration Arbiter Register).

This algorithm grants to each port a percentage of the bandwidth. The NRD_NWD_BDW_Px field defines the percentage allocated to each port.

The percentage of the bandwidth is programmed with the NRD_NWD_BDW_Px fields (see MPDDRC Configuration Arbiter Register).

The TIMEOUT_Px field defines the delay between two accesses on the same port in number of cycles rearbitrating the access to another port. This field allows to avoid a timeout on the system because some masters have the particularity to add idle cycles between two consecutive accesses (see MPDDRC Configuration Arbiter Register).

This algorithm dispatches the requests from different masters to the DDR-SDRAM device in a round-robin manner. If two or more master requests arise at the same time, the master with the lowest number is serviced first, then the others are serviced in a round-robin manner when the allocated bandwidth is reached or when the timeout value is reached.

The BDW_BURST field allows to arbitrate either when the current master reaches exactly the programmed bandwidth, or when the current master reaches exactly the programmed bandwidth and the current access is ended (see MPDDRC Configuration Arbiter Register).

To provide the maximum throughput for the DDR-SDRAM device, arbitration must only take place during the following cycles:

1. Timeout is reached: the delay between two accesses is equal to TIMEOUT_Px.
2. Allocated Bandwidth is reached although the current cycle is not ended.
3. Allocated Bandwidth is reached and the current cycle is the last cycle of a transfer.

### 35.5.6 Scrambling/Unscrambling Function

The external data bus can be scrambled in order to prevent intellectual property data located in off-chip memories from being easily recovered by analyzing data at the package pin level of either the microcontroller or the memory device.

The scrambling and unscrambling are performed on-the-fly without additional wait states.

The scrambling method depends on two user-configurable key registers, KEY1 in the “MPDDRC OCMS KEY1 Register” and KEY2 in the “MPDDRC OCMS KEY2 Register”. These key registers are only accessible in Write mode.

The key must be securely stored in a reliable non-volatile memory in order to recover data from the off-chip memory. Any data scrambled with a given key cannot be recovered if the key is lost.

The scrambling/unscrambling function can be enabled or disabled by programming the "MPDDRC OCMS Register".

### 35.5.7 Register Write Protection

To prevent any single software error from corrupting MPDDRC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the MPDDRC Write Protection Mode Register (MPDDRC_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the MPDDRC Write Protection Status Register (MPDDRC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading MPDDRC_WPSR.

The following registers are write-protected when the bit WPEN is set:

- MPDDRC Mode Register
- MPDDRC Refresh Timer Register
- MPDDRC Configuration Register
- MPDDRC Timing Parameter 0 Register
- MPDDRC Timing Parameter 1 Register
- MPDDRC Memory Device Register
- MPDDRC Low-power DDR2 Low-power DDR3 and DDR3 Calibration and MR4 Register
- MPDDRC OCMS Register
- MPDDRC OCMS KEY1 Register
- MPDDRC OCMS KEY2 Register

### 35.5.8 Monitor

The MPDDRC embeds a monitor which collects bus transaction information from four/eight ports of MPDDRC. This information, such as accumulated latency (MPDDRC Monitor Information Port x Register (TOTAL_LATENCY)) or number of transfers (MPDDRC Monitor Information Port x Register (NB_TRANSFERS)), can be used to calculate the average latency for each port.

Configuration registers ("MPDDRC Monitor Configuration Register", "MPDDRC Monitor Address High/Low Port x Register") are used to define the type of transaction collected (read, write or read/write) and the address range snooped.

Before enabling the monitor, the address range must be defined. This space will be snooped. To enable the monitor, the "EN_MONI: Enable Monitor" bit must be set to 1 and the "RUN: Control Monitor" bit must be set to 1.

## 35.6 Software Interface/SDRAM Organization, Address Mapping

The DDR-SDRAM address space is organized into banks, rows and columns. The MPDDRC maps different memory types depending on values set in the Configuration register (MPDDRC_CR) (see MPDDRC Configuration Register). The tables that follow illustrate the relation between CPU addresses and columns, rows and banks addresses for 16/32-bit memory data bus widths.

The MPDDRC supports address mapping in Linear mode.

Sequential mode is a method for address mapping where banks alternate at each last DDR-SDRAM page of the current bank.

Interleaved mode is a method for address mapping where banks alternate at each DDR-SDRAM end of page of the current bank.

The MPDDRC makes the DDR-SDRAM device access protocol transparent to the user. The tables that follow illustrate the DDR-SDRAM device memory mapping seen by the user in correlation with the device structure. Various configurations are illustrated.

### 35.6.1 DDR-SDRAM Address Mapping for 16-bit Memory Data Bus Width

**Table 35-3. Sequential Mapping for DDR-SDRAM Configuration, 2K Rows, 256/512/1024/2048/4096 Columns, 4 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | Bk[1:0] | | Row[10:0] | | | | | | | | | | | Column[7:0] | | | | | | | | M0 |
| | | | | | Bk[1:0] | | | Row[10:0] | | | | | | | | | | | Column[8:0] | | | | | | | | M0 |
| | | | | Bk[1:0] | | | | Row[10:0] | | | | | | | | | | | Column[9:0] | | | | | | | | M0 |
| | | | Bk[1:0] | | | | | Row[10:0] | | | | | | | | | | | Column[10:0] | | | | | | | | M0 |
| | | Bk[1:0] | | | | | | Row[10:0] | | | | | | | | | | | Column[11:0] | | | | | | | | M0 |

**Table 35-4. Interleaved Mapping for DDR-SDRAM Configuration, 2K Rows, 256/512/1024/2048/4096 Columns, 4 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | Row[10:0] | | | | | | | | | | Bk[1:0] | | Column[7:0] | | | | | | | | M0 |
| | | | | | | Row[10:0] | | | | | | | | | | Bk[1:0] | | | Column[8:0] | | | | | | | | M0 |
| | | | | | Row[10:0] | | | | | | | | | | Bk[1:0] | | | | Column[9:0] | | | | | | | | M0 |
| | | | | Row[10:0] | | | | | | | | | | Bk[1:0] | | | | | Column[10:0] | | | | | | | | M0 |
| | | | Row[10:0] | | | | | | | | | | Bk[1:0] | | | | | | Column[11:0] | | | | | | | | M0 |

**Table 35-5. Sequential Mapping for DDR-SDRAM Configuration: 4K Rows, 256/512/1024/2048/4096 Columns, 4 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | Bk[1:0] | | Row[11:0] | | | | | | | | | | | | Column[7:0] | | | | | | | | M0 |
| | | | | Bk[1:0] | | | Row[11:0] | | | | | | | | | | | | Column[8:0] | | | | | | | | M0 |
| | | | Bk[1:0] | | | | Row[11:0] | | | | | | | | | | | | Column[9:0] | | | | | | | | M0 |
| | | Bk[1:0] | | | | | Row[11:0] | | | | | | | | | | | | Column[10:0] | | | | | | | | M0 |
| | Bk[1:0] | | | | | | Row[11:0] | | | | | | | | | | | | Column[11:0] | | | | | | | | M0 |

**Table 35-6. Interleaved Mapping for DDR-SDRAM Configuration: 4K Rows, 256/512/1024/2048/4096 Columns, 4 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | Row[11:0] | | | | | | | | | | | Bk[1:0] | Column[7:0] | | | | | | | | | M0 |
| | | | | | Row[11:0] | | | | | | | | | | | Bk[1:0] | Column[8:0] | | | | | | | | | | M0 |
| | | | | Row[11:0] | | | | | | | | | | | Bk[1:0] | Column[9:0] | | | | | | | | | | | M0 |
| | | | Row[11:0] | | | | | | | | | | | Bk[1:0] | Column[10:0] | | | | | | | | | | | | M0 |
| | | Row[11:0] | | | | | | | | | | | Bk[1:0] | Column[11:0] | | | | | | | | | | | | | M0 |

**Table 35-7. Sequential Mapping for DDR-SDRAM Configuration: 8K Rows, 512/1024/2048/4096 Columns, 4 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | Bk[1:0] | | Row[12:0] | | | | | | | | | | | | | Column[8:0] | | | | | | | | | M0 |
| | | Bk[1:0] | | | Row[12:0] | | | | | | | | | | | | Column[9:0] | | | | | | | | | | M0 |
| | Bk[1:0] | | | | Row[12:0] | | | | | | | | | | | Column[10:0] | | | | | | | | | | | M0 |
| Bk[1:0] | | | | | Row[12:0] | | | | | | | | | | Column[11:0] | | | | | | | | | | | | M0 |

**Table 35-8. Interleaved Mapping for DDR-SDRAM Configuration: 8K Rows, 512/1024/2048/4096 Columns, 4 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | Row[12:0] | | | | | | | | | | | | | Bk[1:0] | | Column[8:0] | | | | | | | | | M0 |
| | | Row[12:0] | | | | | | | | | | | | | Bk[1:0] | | Column[9:0] | | | | | | | | | | M0 |
| | Row[12:0] | | | | | | | | | | | | | Bk[1:0] | | Column[10:0] | | | | | | | | | | | M0 |
| Row[12:0] | | | | | | | | | | | | | Bk[1:0] | | Column[11:0] | | | | | | | | | | | | M0 |

**Table 35-9. Sequential Mapping for DDR-SDRAM Configuration: 16K Rows, 512/1024/2048 Columns, 4 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | Bk[1:0] | | Row[13:0] | | | | | | | | | | | | | | Column[8:0] | | | | | | | | | M0 |
| | Bk[1:0] | | Row[13:0] | | | | | | | | | | | | | | Column[9:0] | | | | | | | | | | M0 |
| Bk[1:0] | | Row[13:0] | | | | | | | | | | | | | | Column[10:0] | | | | | | | | | | | M0 |

**Table 35-10. Interleaved Mapping for DDR-SDRAM Configuration: 16K Rows, 512/1024/2048 Columns, 4 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | Row[13:0] | | | | | | | | | | | | | | Bk[1:0] | | Column[8:0] | | | | | | | | | M0 |
| | Row[13:0] | | | | | | | | | | | | | | Bk[1:0] | | | Column[9:0] | | | | | | | | | M0 |
| Row[13:0] | | | | | | | | | | | | | | Bk[1:0] | | | | Column[10:0] | | | | | | | | | M0 |

**Table 35-11. Sequential Mapping for DDR-SDRAM Configuration: 8K Rows,1024/ Columns, 8 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | Bk[2:0] | | | Row[12:0] | | | | | | | | | | | | | Column[9:0] | | | | | | | | | | M0 |

**Table 35-12. Interleaved Mapping for DDR-SDRAM Configuration: 8K Rows,1024/ Columns, 8 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | Row[12:0] | | | | | | | | | | | | | Bk[2:0] | | | Column[9:0] | | | | | | | | | | M0 |

**Table 35-13. Sequential Mapping for DDR-SDRAM Configuration: 16K Rows,1024/ Columns, 8 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Bk[2:0] | | | Row[13:0] | | | | | | | | | | | | | | Column[9:0] | | | | | | | | | | M0 |

**Table 35-14. Interleaved Mapping for DDR-SDRAM Configuration: 16K Rows,1024/ Columns, 8 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Row[13:0] | | | | | | | | | | | | | | Bk[2:0] | | | Column[9:0] | | | | | | | | | | M0 |

### 35.6.2 DDR-SDRAM Address Mapping for 32-bit Memory Data Bus Width

**Table 35-15. Sequential Mapping DDR-SDRAM Configuration Mapping: 2K Rows, 512/1024/2048 Columns, 4 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | Bk[1:0] | | Row[10:0] | | | | | | | | | | | Column[8:0] | | | | | | | | | M[1:0] | |

**.....continued**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | Bk[1:0] | | Row[10:0] | | | | | | | | | | | Column[9:0] | | | | | | | | | | M[1:0] | |
| | | | Bk[1:0] | | Row[10:0] | | | | | | | | | | | Column[10:0] | | | | | | | | | | | M[1:0] | |

**Table 35-16. Interleaved Mapping DDR-SDRAM Configuration Mapping: 2K Rows, 512/1024/2048 Columns, 4 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | Row[10:0] | | | | | | | | | | | Bk[1:0] | | Column[8:0] | | | | | | | | | M[1:0] | |
| | | | | Row[10:0] | | | | | | | | | | | Bk[1:0] | | Column[9:0] | | | | | | | | | | M[1:0] | |
| | | | Row[10:0] | | | | | | | | | | | Bk[1:0] | | Column[10:0] | | | | | | | | | | | M[1:0] | |

**Table 35-17. Sequential Mapping DDR-SDRAM Configuration Mapping: 4K Rows, 256/512/1024/2048 Columns, 4 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | Bk[1:0] | | Row[11:0] | | | | | | | | | | | | Column[7:0] | | | | | | | | M[1:0] | |
| | | | | Bk[1:0] | | Row[11:0] | | | | | | | | | | | | Column[8:0] | | | | | | | | | M[1:0] | |
| | | | Bk[1:0] | | Row[11:0] | | | | | | | | | | | | Column[9:0] | | | | | | | | | | M[1:0] | |
| | | Bk[1:0] | | Row[11:0] | | | | | | | | | | | | Column[10:0] | | | | | | | | | | | M[1:0] | |

**Table 35-18. Interleaved Mapping DDR-SDRAM Configuration Mapping: 4K Rows, 256/512/1024/2048 Columns, 4 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | Row[11:0] | | | | | | | | | | | | Bk[1:0] | | Column[7:0] | | | | | | | | M[1:0] | |
| | | | | Row[11:0] | | | | | | | | | | | | Bk[1:0] | | Column[8:0] | | | | | | | | | M[1:0] | |
| | | | Row[11:0] | | | | | | | | | | | | Bk[1:0] | | Column[9:0] | | | | | | | | | | M[1:0] | |
| | | Row[11:0] | | | | | | | | | | | | Bk[1:0] | | Column[10:0] | | | | | | | | | | | M[1:0] | |

**Table 35-19. Sequential Mapping DDR-SDRAM Configuration Mapping: 8K Rows, 512/1024/2048 Columns, 4 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | Bk[1:0] | | Row[12:0] | | | | | | | | | | | | | Column[8:0] | | | | | | | | | M[1:0] | |
| | | Bk[1:0] | | | Row[12:0] | | | | | | | | | | | | | Column[9:0] | | | | | | | | | M[1:0] | |
| | Bk[1:0] | | | | Row[12:0] | | | | | | | | | | | | | Column[10:0] | | | | | | | | | M[1:0] | |

**Table 35-20. Interleaved Mapping DDR-SDRAM Configuration Mapping: 8K Rows, 512/1024/2048 Columns, 4 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | Row[13:0] | | | | | | | | | | | | | Bk[1:0] | | Column[8:0] | | | | | | | | | M[1:0] | |
| | | Row[13:0] | | | | | | | | | | | | | Bk[1:0] | | | Column[9:0] | | | | | | | | | M[1:0] | |
| | Row[13:0] | | | | | | | | | | | | | Bk[1:0] | | | | Column[10:0] | | | | | | | | | M[1:0] | |

**Table 35-21. Sequential Mapping DDR-SDRAM Configuration Mapping: 8K Rows, 512/1024 Columns, 8 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | Bk[2:0] | | | Row[12:0] | | | | | | | | | | | | | Column[8:0] | | | | | | | | | M[1:0] | |
| | Bk[2:0] | | | | Row[12:0] | | | | | | | | | | | | | Column[9:0] | | | | | | | | | M[1:0] | |

**Table 35-22. Interleaved Mapping DDR-SDRAM Configuration Mapping: 8K Rows, 512/1024 Columns, 8 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | Row[12:0] | | | | | | | | | | | | | Bk[2:0] | | | Column[8:0] | | | | | | | | | M[1:0] | |
| | Row[12:0] | | | | | | | | | | | | | Bk[2:0] | | | | Column[9:0] | | | | | | | | | M[1:0] | |

**Table 35-23. Sequential Mapping DDR-SDRAM Configuration Mapping: 16K Rows, 1024 Columns, 4 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | Bk[1:0] | | Row[13:0] | | | | | | | | | | | | | | Column[9:0] | | | | | | | | | | M[1:0] | |

**Table 35-24. Interleaved Mapping DDR-SDRAM Configuration Mapping: 16K Rows, 1024 Columns, 4 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | Row[13:0] | | | | | | | | | | | | | | Bk[1:0] | | Column[9:0] | | | | | | | | | | M[1:0] | |

**Table 35-25. Sequential Mapping DDR-SDRAM Configuration Mapping: 16K Rows, 1024 Columns, 8 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Bk[2:0] | | | Row[13:0] | | | | | | | | | | | | | | Column[9:0] | | | | | | | | | | M[1:0] | |

**Table 35-26. Interleaved Mapping DDR-SDRAM Configuration Mapping: 16K Rows, 1024 Columns, 8 banks**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Row[13:0] | | | | | | | | | | | | | | Bk[2:0] | | | Column[9:0] | | | | | | | | | | M[1:0] | |

### 35.6.3 DDR-SDRAM Address Mapping for Low-cost Memories

**Table 35-27. Sequential Mapping for DDR-SDRAM Configuration, 2K Rows, 512 Columns, 2 banks, 16 bits**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | Bk | Row[10:0] | | | | | | | | | | | Column[8:0] | | | | | | | | | M0 |

**Table 35-28. Interleaved Mapping for DDR-SDRAM Configuration, 2K Rows, 512 Columns, 2 banks, 16 bits**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | Row[10:0] | | | | | | | | | | | Bk | Column[8:0] | | | | | | | | | M0 |

**Table 35-29. Sequential Mapping for DDR-SDRAM Configuration: 4K Rows, 256 Columns, 2 banks, 32 bits**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | Bk | Row[11:0] | | | | | | | | | | | Column[7:0] | | | | | | | | | M[1:0] |

**Table 35-30. Interleaved Mapping for DDR-SDRAM Configuration: 4K Rows, 256 Columns, 2 banks, 32 bits**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | Row[11:0] | | | | | | | | | | | Bk | Column[7:0] | | | | | | | | M[1:0] | |

**Note:**

1. M[1:0] is the byte address inside a 32-bit word.
2. Bk[2] = BA2, Bk[1] = BA1, Bk[0] = BA0

## 35.7 Register Summary

The User Interface is connected to the APB bus. The MPDDRC is programmed using the registers listed in the following table.

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | MPDDRC_MR | 7:0 | | | | DAI | | MODE[2:0] | | |
| | | 15:8 | MRS[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | MPDDRC_RTR | 7:0 | COUNT[7:0] | | | | | | | |
| | | 15:8 | | | | | COUNT[11:8] | | | |
| | | 23:16 | | MR4_VALUE[2:0] | | | | | REF_PB | ADJ_REF |
| | | 31:24 | | | | | | | | |
| 0x08 | MPDDRC_CR | 7:0 | DLL | CAS[2:0] | | | NR[1:0] | | NC[1:0] | |
| | | 15:8 | | OCD[2:0] | | | ZQ[1:0] | | DIS_DLL | DIC_DS |
| | | 23:16 | UNAL | DECOD | NDQS | NB | LC_LPDDR1 | | ENRDM | DQMS |
| | | 31:24 | | | | | | | | |
| 0x0C | MPDDRC_TPR0 | 7:0 | TRCD[3:0] | | | | TRAS[3:0] | | | |
| | | 15:8 | TRC[3:0] | | | | TWR[3:0] | | | |
| | | 23:16 | TRRD[3:0] | | | | TRP[3:0] | | | |
| | | 31:24 | TMRD[3:0] | | | | | TWTR[2:0] | | |
| 0x10 | MPDDRC_TPR1 | 7:0 | | TRFC[6:0] | | | | | | |
| | | 15:8 | TXSNR[7:0] | | | | | | | |
| | | 23:16 | TXSRD[7:0] | | | | | | | |
| | | 31:24 | | | | | TXP[3:0] | | | |
| 0x14 | MPDDRC_TPR2 | 7:0 | TXARDS[3:0] | | | | TXARD[3:0] | | | |
| | | 15:8 | | TRTP[2:0] | | | TRPA[3:0] | | | |
| | | 23:16 | | | | | TFAW[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x18 ... 0x1B | Reserved | | | | | | | | | |
| 0x1C | MPDDRC_LPR | 7:0 | | PASR[2:0] | | | LPDDR2_LPDDR3_PWOFF | CLK_FR | LPCB[1:0] | |
| | | 15:8 | | | TIMEOUT[1:0] | | | DS[2:0] | | |
| | | 23:16 | | | UPD_MR[1:0] | | | | | APDE |
| | | 31:24 | | | | | | | SELF_DONE | CHG_FRQ |
| 0x20 | MPDDRC_MD | 7:0 | RL3 | WL | | DBW | | MD[2:0] | | |
| | | 15:8 | MANU_ID[7:0] | | | | | | | |
| | | 23:16 | REV_ID[7:0] | | | | | | | |
| | | 31:24 | IO_WIDTH[1:0] | | DENSITY[3:0] | | | | TYPE[1:0] | |
| 0x24 ... 0x27 | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x28 | MPDDRC_LPDDR23_LPR | 7:0 | BK_MASK_PASR[7:0] | | | | | | | |
| | | 15:8 | SEG_MASK[7:0] | | | | | | | |
| | | 23:16 | SEG_MASK[15:8] | | | | | | | |
| | | 31:24 | | | | | DS[3:0] | | | |
| 0x2C | MPDDRC_LPDDR2_LPDDR3_DDR3_CAL_MR4 | 7:0 | COUNT_CAL[7:0] | | | | | | | |
| | | 15:8 | COUNT_CAL[15:8] | | | | | | | |
| | | 23:16 | MR4_READ[7:0] | | | | | | | |
| | | 31:24 | MR4_READ[15:8] | | | | | | | |
| 0x30 | MPDDRC_LPDDR2_LPDDR3_DDR3_TIM_CAL | 7:0 | ZQCS[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | RZQI[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x34 | MPDDRC_IO_CALIBR | 7:0 | | | | EN_CALIB | | RDIV[2:0] | | |
| | | 15:8 | | TZQIO[6:0] | | | | | | |
| | | 23:16 | CALCODEN[3:0] | | | | CALCODEP[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x38 | MPDDRC_OCMS | 7:0 | | | | | | | | SCR_EN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x3C | MPDDRC_OCMS_KEY1 | 7:0 | KEY1[7:0] | | | | | | | |
| | | 15:8 | KEY1[15:8] | | | | | | | |
| | | 23:16 | KEY1[23:16] | | | | | | | |
| | | 31:24 | KEY1[31:24] | | | | | | | |
| 0x40 | MPDDRC_OCMS_KEY2 | 7:0 | KEY2[7:0] | | | | | | | |
| | | 15:8 | KEY2[15:8] | | | | | | | |
| | | 23:16 | KEY2[23:16] | | | | | | | |
| | | 31:24 | KEY2[31:24] | | | | | | | |
| 0x44 | MPDDRC_CONF_ARBITER | 7:0 | | | | | BDW_MAX_CUR | | ARB[1:0] | |
| | | 15:8 | RQ_WD_P7 | RQ_WD_P6 | RQ_WD_P5 | RQ_WD_P4 | RQ_WD_P3 | RQ_WD_P2 | RQ_WD_P1 | RQ_WD_P0 |
| | | 23:16 | MA_PR_P7 | MA_PR_P6 | MA_PR_P5 | MA_PR_P4 | MA_PR_P3 | MA_PR_P2 | MA_PR_P1 | MA_PR_P0 |
| | | 31:24 | BDW_BURST_P7 | BDW_BURST_P6 | BDW_BURST_P5 | BDW_BURST_P4 | BDW_BURST_P3 | BDW_BURST_P2 | BDW_BURST_P1 | BDW_BURST_P0 |
| 0x48 | MPDDRC_TIMEOUT | 7:0 | TIMEOUT_P1[3:0] | | | | TIMEOUT_P0[3:0] | | | |
| | | 15:8 | TIMEOUT_P3[3:0] | | | | TIMEOUT_P2[3:0] | | | |
| | | 23:16 | TIMEOUT_P5[3:0] | | | | TIMEOUT_P4[3:0] | | | |
| | | 31:24 | TIMEOUT_P7[3:0] | | | | TIMEOUT_P6[3:0] | | | |
| 0x4C | MPDDRC_REQ_PORT_0123 | 7:0 | NRQ_NWD_BDW_P0[7:0] | | | | | | | |
| | | 15:8 | NRQ_NWD_BDW_P1[7:0] | | | | | | | |
| | | 23:16 | NRQ_NWD_BDW_P2[7:0] | | | | | | | |
| | | 31:24 | NRQ_NWD_BDW_P3[7:0] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x50 | MPDDRC_REQ_PORT_4567 | 7:0 | NRQ_NWD_BDW_P4[7:0] | | | | | | | | |
| | | 15:8 | NRQ_NWD_BDW_P5[7:0] | | | | | | | | |
| | | 23:16 | NRQ_NWD_BDW_P6[7:0] | | | | | | | | |
| | | 31:24 | NRQ_NWD_BDW_P7[7:0] | | | | | | | | |
| 0x54 | MPDDRC_BDW_PORT_0123 | 7:0 | | BDW_P0[6:0] | | | | | | | |
| | | 15:8 | | BDW_P1[6:0] | | | | | | | |
| | | 23:16 | | BDW_P2[6:0] | | | | | | | |
| | | 31:24 | | BDW_P3[6:0] | | | | | | | |
| 0x58 | MPDDRC_BDW_PORT_4567 | 7:0 | BDW_P4[7:0] | | | | | | | | |
| | | 15:8 | BDW_P5[7:0] | | | | | | | | |
| | | 23:16 | BDW_P6[7:0] | | | | | | | | |
| | | 31:24 | BDW_P7[7:0] | | | | | | | | |
| 0x5C | MPDDRC_RD_DATA_PATH | 7:0 | | | | | | | SHIFT_SAMPLING[1:0] | | |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x60 | MPDDRC_MCFGR | 7:0 | | | | RUN | | | SOFT_RESET | EN_MONI | |
| | | 15:8 | | | | INFO[1:0] | | REFR_CALIB | READ_WRITE[1:0] | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x64 | MPDDRC_MADDR0 | 7:0 | ADDR_LOW_PORT0[7:0] | | | | | | | | |
| | | 15:8 | ADDR_LOW_PORT0[15:8] | | | | | | | | |
| | | 23:16 | ADDR_HIGH_PORT0[7:0] | | | | | | | | |
| | | 31:24 | ADDR_HIGH_PORT0[15:8] | | | | | | | | |
| 0x68 | MPDDRC_MADDR1 | 7:0 | ADDR_LOW_PORT1[7:0] | | | | | | | | |
| | | 15:8 | ADDR_LOW_PORT1[15:8] | | | | | | | | |
| | | 23:16 | ADDR_HIGH_PORT1[7:0] | | | | | | | | |
| | | 31:24 | ADDR_HIGH_PORT1[15:8] | | | | | | | | |
| 0x6C | MPDDRC_MADDR2 | 7:0 | ADDR_LOW_PORT2[7:0] | | | | | | | | |
| | | 15:8 | ADDR_LOW_PORT2[15:8] | | | | | | | | |
| | | 23:16 | ADDR_HIGH_PORT2[7:0] | | | | | | | | |
| | | 31:24 | ADDR_HIGH_PORT2[15:8] | | | | | | | | |
| 0x70 | MPDDRC_MADDR3 | 7:0 | ADDR_LOW_PORT3[7:0] | | | | | | | | |
| | | 15:8 | ADDR_LOW_PORT3[15:8] | | | | | | | | |
| | | 23:16 | ADDR_HIGH_PORT3[7:0] | | | | | | | | |
| | | 31:24 | ADDR_HIGH_PORT3[15:8] | | | | | | | | |
| 0x74 | MPDDRC_MADDR4 | 7:0 | ADDR_LOW_PORT4[7:0] | | | | | | | | |
| | | 15:8 | ADDR_LOW_PORT4[15:8] | | | | | | | | |
| | | 23:16 | ADDR_HIGH_PORT4[7:0] | | | | | | | | |
| | | 31:24 | ADDR_HIGH_PORT4[15:8] | | | | | | | | |
| 0x78 | MPDDRC_MADDR5 | 7:0 | ADDR_LOW_PORT5[7:0] | | | | | | | | |
| | | 15:8 | ADDR_LOW_PORT5[15:8] | | | | | | | | |
| | | 23:16 | ADDR_HIGH_PORT5[7:0] | | | | | | | | |
| | | 31:24 | ADDR_HIGH_PORT5[15:8] | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x7C | MPDDRC_MADDR6 | 7:0 | ADDR_LOW_PORT6[7:0] | | | | | | | |
| | | 15:8 | ADDR_LOW_PORT6[15:8] | | | | | | | |
| | | 23:16 | ADDR_HIGH_PORT6[7:0] | | | | | | | |
| | | 31:24 | ADDR_HIGH_PORT6[15:8] | | | | | | | |
| 0x80 | MPDDRC_MADDR7 | 7:0 | ADDR_LOW_PORT7[7:0] | | | | | | | |
| | | 15:8 | ADDR_LOW_PORT7[15:8] | | | | | | | |
| | | 23:16 | ADDR_HIGH_PORT7[7:0] | | | | | | | |
| | | 31:24 | ADDR_HIGH_PORT7[15:8] | | | | | | | |
| 0x84 | MPDDRC_MINFO0 (MAX_WAIT) | 7:0 | MAX_PORT0_WAITING[7:0] | | | | | | | |
| | | 15:8 | MAX_PORT0_WAITING[15:8] | | | | | | | |
| | | 23:16 | | SIZE[2:0] | | | | BURST[2:0] | | |
| | | 31:24 | | | | | | | | READ_WRITE |
| 0x84 | MPDDRC_MINFO0 (NB_TRANSFERS) | 7:0 | P0_NB_TRANSFERS[7:0] | | | | | | | |
| | | 15:8 | P0_NB_TRANSFERS[15:8] | | | | | | | |
| | | 23:16 | P0_NB_TRANSFERS[23:16] | | | | | | | |
| | | 31:24 | P0_NB_TRANSFERS[31:24] | | | | | | | |
| 0x84 | MPDDRC_MINFO0 (TOTAL_LATENCY) | 7:0 | P0_TOTAL_LATENCY[7:0] | | | | | | | |
| | | 15:8 | P0_TOTAL_LATENCY[15:8] | | | | | | | |
| | | 23:16 | P0_TOTAL_LATENCY[23:16] | | | | | | | |
| | | 31:24 | P0_TOTAL_LATENCY[31:24] | | | | | | | |
| 0x88 | MPDDRC_MINFO1 (MAX_WAIT) | 7:0 | MAX_PORT1_WAITING[7:0] | | | | | | | |
| | | 15:8 | MAX_PORT1_WAITING[15:8] | | | | | | | |
| | | 23:16 | | SIZE[2:0] | | | | BURST[2:0] | | |
| | | 31:24 | | | | | | | | READ_WRITE |
| 0x88 | MPDDRC_MINFO1 (NB_TRANSFERS) | 7:0 | P1_NB_TRANSFERS[7:0] | | | | | | | |
| | | 15:8 | P1_NB_TRANSFERS[15:8] | | | | | | | |
| | | 23:16 | P1_NB_TRANSFERS[23:16] | | | | | | | |
| | | 31:24 | P1_NB_TRANSFERS[31:24] | | | | | | | |
| 0x88 | MPDDRC_MINFO1 (TOTAL_LATENCY) | 7:0 | P1_TOTAL_LATENCY[7:0] | | | | | | | |
| | | 15:8 | P1_TOTAL_LATENCY[15:8] | | | | | | | |
| | | 23:16 | P1_TOTAL_LATENCY[23:16] | | | | | | | |
| | | 31:24 | P1_TOTAL_LATENCY[31:24] | | | | | | | |
| 0x8C | MPDDRC_MINFO2 (MAX_WAIT) | 7:0 | MAX_PORT2_WAITING[7:0] | | | | | | | |
| | | 15:8 | MAX_PORT2_WAITING[15:8] | | | | | | | |
| | | 23:16 | | SIZE[2:0] | | | | BURST[2:0] | | |
| | | 31:24 | | | | | | | | READ_WRITE |
| 0x8C | MPDDRC_MINFO2 (NB_TRANSFERS) | 7:0 | P2_NB_TRANSFERS[7:0] | | | | | | | |
| | | 15:8 | P2_NB_TRANSFERS[15:8] | | | | | | | |
| | | 23:16 | P2_NB_TRANSFERS[23:16] | | | | | | | |
| | | 31:24 | P2_NB_TRANSFERS[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x8C | MPDDRC_MINFO2 (TOTAL_LATENCY) | 7:0 | P2_TOTAL_LATENCY[7:0] | | | | | | | |
| | | 15:8 | P2_TOTAL_LATENCY[15:8] | | | | | | | |
| | | 23:16 | P2_TOTAL_LATENCY[23:16] | | | | | | | |
| | | 31:24 | P2_TOTAL_LATENCY[31:24] | | | | | | | |
| 0x90 | MPDDRC_MINFO3 (MAX_WAIT) | 7:0 | MAX_PORT3_WAITING[7:0] | | | | | | | |
| | | 15:8 | MAX_PORT3_WAITING[15:8] | | | | | | | |
| | | 23:16 | | SIZE[2:0] | | | | BURST[2:0] | | |
| | | 31:24 | | | | | | | | READ_WRITE |
| 0x90 | MPDDRC_MINFO3 (NB_TRANSFERS) | 7:0 | P3_NB_TRANSFERS[7:0] | | | | | | | |
| | | 15:8 | P3_NB_TRANSFERS[15:8] | | | | | | | |
| | | 23:16 | P3_NB_TRANSFERS[23:16] | | | | | | | |
| | | 31:24 | P3_NB_TRANSFERS[31:24] | | | | | | | |
| 0x90 | MPDDRC_MINFO3 (TOTAL_LATENCY) | 7:0 | P3_TOTAL_LATENCY[7:0] | | | | | | | |
| | | 15:8 | P3_TOTAL_LATENCY[15:8] | | | | | | | |
| | | 23:16 | P3_TOTAL_LATENCY[23:16] | | | | | | | |
| | | 31:24 | P3_TOTAL_LATENCY[31:24] | | | | | | | |
| 0x94 | MPDDRC_MINFO4 (MAX_WAIT) | 7:0 | MAX_PORT4_WAITING[7:0] | | | | | | | |
| | | 15:8 | MAX_PORT4_WAITING[15:8] | | | | | | | |
| | | 23:16 | | SIZE[2:0] | | | | BURST[2:0] | | |
| | | 31:24 | | | | | | | | READ_WRITE |
| 0x94 | MPDDRC_MINFO4 (NB_TRANSFERS) | 7:0 | P4_NB_TRANSFERS[7:0] | | | | | | | |
| | | 15:8 | P4_NB_TRANSFERS[15:8] | | | | | | | |
| | | 23:16 | P4_NB_TRANSFERS[23:16] | | | | | | | |
| | | 31:24 | P4_NB_TRANSFERS[31:24] | | | | | | | |
| 0x94 | MPDDRC_MINFO4 (TOTAL_LATENCY) | 7:0 | P4_TOTAL_LATENCY[7:0] | | | | | | | |
| | | 15:8 | P4_TOTAL_LATENCY[15:8] | | | | | | | |
| | | 23:16 | P4_TOTAL_LATENCY[23:16] | | | | | | | |
| | | 31:24 | P4_TOTAL_LATENCY[31:24] | | | | | | | |
| 0x98 | MPDDRC_MINFO5 (MAX_WAIT) | 7:0 | MAX_PORT5_WAITING[7:0] | | | | | | | |
| | | 15:8 | MAX_PORT5_WAITING[15:8] | | | | | | | |
| | | 23:16 | | SIZE[2:0] | | | | BURST[2:0] | | |
| | | 31:24 | | | | | | | | READ_WRITE |
| 0x98 | MPDDRC_MINFO5 (NB_TRANSFERS) | 7:0 | P5_NB_TRANSFERS[7:0] | | | | | | | |
| | | 15:8 | P5_NB_TRANSFERS[15:8] | | | | | | | |
| | | 23:16 | P5_NB_TRANSFERS[23:16] | | | | | | | |
| | | 31:24 | P5_NB_TRANSFERS[31:24] | | | | | | | |
| 0x98 | MPDDRC_MINFO5 (TOTAL_LATENCY) | 7:0 | P5_TOTAL_LATENCY[7:0] | | | | | | | |
| | | 15:8 | P5_TOTAL_LATENCY[15:8] | | | | | | | |
| | | 23:16 | P5_TOTAL_LATENCY[23:16] | | | | | | | |
| | | 31:24 | P5_TOTAL_LATENCY[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x9C | MPDDRC_MINFO6 (MAX_WAIT) | 7:0 | MAX_PORT6_WAITING[7:0] | | | | | | | |
| | | 15:8 | MAX_PORT6_WAITING[15:8] | | | | | | | |
| | | 23:16 | | SIZE[2:0] | | | | BURST[2:0] | | |
| | | 31:24 | | | | | | | | READ_WRITE |
| 0x9C | MPDDRC_MINFO6 (NB_TRANSFERS) | 7:0 | P6_NB_TRANSFERS[7:0] | | | | | | | |
| | | 15:8 | P6_NB_TRANSFERS[15:8] | | | | | | | |
| | | 23:16 | P6_NB_TRANSFERS[23:16] | | | | | | | |
| | | 31:24 | P6_NB_TRANSFERS[31:24] | | | | | | | |
| 0x9C | MPDDRC_MINFO6 (TOTAL_LATENCY) | 7:0 | P6_TOTAL_LATENCY[7:0] | | | | | | | |
| | | 15:8 | P6_TOTAL_LATENCY[15:8] | | | | | | | |
| | | 23:16 | P6_TOTAL_LATENCY[23:16] | | | | | | | |
| | | 31:24 | P6_TOTAL_LATENCY[31:24] | | | | | | | |
| 0xA0 | MPDDRC_MINFO7 (MAX_WAIT) | 7:0 | MAX_PORT7_WAITING[7:0] | | | | | | | |
| | | 15:8 | MAX_PORT7_WAITING[15:8] | | | | | | | |
| | | 23:16 | | SIZE[2:0] | | | | BURST[2:0] | | |
| | | 31:24 | | | | | | | | READ_WRITE |
| 0xA0 | MPDDRC_MINFO7 (NB_TRANSFERS) | 7:0 | P7_NB_TRANSFERS[7:0] | | | | | | | |
| | | 15:8 | P7_NB_TRANSFERS[15:8] | | | | | | | |
| | | 23:16 | P7_NB_TRANSFERS[23:16] | | | | | | | |
| | | 31:24 | P7_NB_TRANSFERS[31:24] | | | | | | | |
| 0xA0 | MPDDRC_MINFO7 (TOTAL_LATENCY) | 7:0 | P7_TOTAL_LATENCY[7:0] | | | | | | | |
| | | 15:8 | P7_TOTAL_LATENCY[15:8] | | | | | | | |
| | | 23:16 | P7_TOTAL_LATENCY[23:16] | | | | | | | |
| | | 31:24 | P7_TOTAL_LATENCY[31:24] | | | | | | | |
| 0xA4 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | MPDDRC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | MPDDRC_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |

### 35.7.1 MPDDRC Mode Register

**Name:** MPDDRC_MR
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MRS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | DAI | | MODE[2:0] | | |
| Access | | | | R | | R/W | R/W | R/W |
| Reset | | | | 0 | | 0 | 0 | 0 |

**Bits 15:8 – MRS[7:0]** Mode Register Select LPDDR2/LPDDR3
Configure this 8-bit field to program all mode registers included in the low-power DDR2-SDRAM device. This field is unique to the low-power DDR2-SDRAM devices and low-power DDR3-SDRAM devices.

**Bit 4 – DAI** Device Auto-initialization Status
This field reports when the device auto-initialization is complete. When Backup mode is used, this information is lost after Backup mode exit.

| Value | Name | Description |
|---|---|---|
| 0 | DAI_COMPLETE | DAI complete |
| 1 | DAI_IN_PROGESSS | DAI still in progress |

**Bits 2:0 – MODE[2:0]** MPDDRC Command Mode
This field defines the command issued by the MPDDRC when the SDRAM device is accessed. This register is used to initialize the SDRAM device and to activate Deep Powerdown mode.

| Value | Name | Description |
|---|---|---|
| 0 | NORMAL_CMD | Normal Mode. Any access to the MPDDRC is decoded normally. To activate this mode, the command must be followed by a write to the DDR-SDRAM. |

| Value | Name | Description |
|---|---|---|
| 1 | NOP_CMD | The MPDDRC issues a NOP command when the DDR-SDRAM device is accessed regardless of the cycle. To activate this mode, the command must be followed by a write to the DDR-SDRAM. |
| 2 | PRCGALL_CMD | The MPDDRC issues the All Banks Precharge command when the DDR-SDRAM device is accessed regardless of the cycle. To activate this mode, the command must be followed by a write to the SDRAM. |
| 3 | LMR_CMD | The MPDDRC issues a Load Mode Register command when the DDR-SDRAM device is accessed regardless of the cycle. To activate this mode, the command must be followed by a write to the DDR-SDRAM. |
| 4 | RFSH_CMD | The MPDDRC issues an Autorefresh command when the DDR-SDRAM device is accessed regardless of the cycle. Previously, an All Banks Precharge command must be issued. To activate this mode, the command must be followed by a write to the DDR-SDRAM. |
| 5 | EXT_LMR_CMD | The MPDDRC issues an Extended Load Mode Register command when the SDRAM device is accessed regardless of the cycle. To activate this mode, the command must be followed by a write to the DDR-SDRAM. The write in the DDR-SDRAM must be done in the appropriate bank. |
| 6 | DEEP_CALIB_MD | Deep Power mode: Access to Deep Powerdown mode<br>Calibration command: to calibrate RTT and RON values for the Process Voltage Temperature (PVT) (DDR3-SDRAM device) |
| 7 | LPDDR2_LPDDR3_CMD | The MPDDRC issues an LPDDR2/LPDDR3 Mode Register command when the device is accessed regardless of the cycle. To activate this mode, the Mode Register command must be followed by a write to the low-power DDR2-SDRAM or to the low-power DDR3-SDRAM. |

### 35.7.2 MPDDRC Refresh Timer Register

**Name:** MPDDRC_RTR
**Offset:** 0x04
**Reset:** 0x00300000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | MR4_VALUE[2:0] | | | | | REF_PB | ADJ_REF |
| Access | | R | R | R | | | R/W | R/W |
| Reset | | 0 | 1 | 1 | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | COUNT[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 22:20 – MR4_VALUE[2:0]** Content of MR4 Register
This field gives the content of MR4 register. This field is updated when MRR command is generated and Adjust Refresh.

Rate bit is enabled. Update is done when read value is different from MR4_VALUE.

This mode is unique to the low-power DDR2-SDRAM devices and LPDDR3 JEDEC memory standards impose derating LPDDR2/LPDDR3 AC timings ($t_{RCD}$, $t_{RC}$, $t_{RAS}$, $t_{RP}$ and $t_{RRD}$) when the value of MR4 is equal to 6. If the application needs to work in extreme conditions, the derating value must be added to AC timings before the power up sequence.

This mode is unique to the low-power DDR2-SDRAM devices and low-power DDR3-SDRAM devices.

**Bit 17 – REF_PB** Refresh Per Bank
This mode is unique to the low-power DDR2-SDRAM devices and low-power DDR3-SDRAM devices.

| Value | Description |
|---|---|
| 0 | Refresh all banks during autorefresh operation. |
| 1 | Refresh the scheduled bank by the bank counter in the memory interface. |

**Bit 16 – ADJ_REF** Adjust Refresh Rate
This mode is unique to the low-power DDR2-SDRAM devices and low-power DDR3-SDRAM devices.

| Value | Description |
|---|---|
| 0 | Adjust refresh rate is not enabled. |
| 1 | Adjust refresh rate is enabled. |

**Bits 11:0 – COUNT[11:0]** MPDDRC Refresh Timer Count
This 12-bit field is loaded into a timer which generates the refresh pulse. Each time the refresh pulse is generated, a refresh sequence is initiated.

The SDRAM requires Auto-refresh cycles at an average periodic interval of $T_{refi}$. The value to be loaded depends on the MPDDRC clock frequency MCK (Master Clock) and average periodic interval of $T_{refi}$.

For example, for an SDRAM with $T_{refi}$ = 7.8 µs and a 133 MHz (7.5 ns) Master clock, the value of the COUNT field is configured: $((7.8 \times 10^{-6}) / (7.5 \times 10^{-9})) = 1040$ or 0x0410.

Low-power DDR2-SDRAM and low-power DDR3-SDRAM devices support Per Bank Refresh operation. In this configuration, average time between refresh command is 0.975 µs. The value of the COUNT field is configured depending on this value. For example, the value of a 133 MHz Master clock refresh timer is 130 or 0x82.

### 35.7.3 MPDDRC Configuration Register

**Name:** MPDDRC_CR
**Offset:** 0x08
**Reset:** 0x00207024
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UNAL | DECOD | NDQS | NB | LC_LPDDR1 | | ENRDM | DQMS |
| Access | R/W | R/W | R/W | R/W | R/W | | R/W | R/W |
| Reset | 0 | 0 | 1 | 0 | 0 | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | OCD[2:0] | | | ZQ[1:0] | | DIS_DLL | DIC_DS |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 1 | 1 | 1 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DLL | CAS[2:0] | | | NR[1:0] | | NC[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |

**Bit 23 – UNAL** This bit must always be written to 1

**Bit 22 – DECOD** Type of Decoding

| Value | Name | Description |
|---|---|---|
| 0 | SEQUENTIAL | Method for address mapping where banks alternate at each last DDR-SDRAM page of the current bank. |
| 1 | INTERLEAVED | Method for address mapping where banks alternate at each DDR-SDRAM end of page of the current bank. |

**Bit 21 – NDQS** Not DQS.
This bit is found only in the DDR2-SDRAM devices.

| Value | Name | Description |
|---|---|---|
| 0 | ENABLED | Not DQS is enabled |
| 1 | DISABLED | Not DQS is disabled |

**Bit 20 – NB** Number of Banks
LC_LPDDR1 is set to 1, NB is not relevant.

| Value | Name | Description |
|---|---|---|
| 0 | 4_BANKS | 4-bank memory devices |
| 1 | 8_BANKS | 8 banks. Only possible when using DDR2-SDRAM and low-power DDR2-SDRAM and DDR3-SDRAM and low-power DDR3-SDRAM devices. |

**Bit 19 – LC_LPDDR1** Low-cost Low-power DDR1

| Value | Name | Description |
|---|---|---|
| 0 | NOT_2_BANKS | Any type of memory devices except of low cost, low density Low Power DDR1. |
| 1 | 2_BANKS_LPDDR1 | Low-cost and low-density low-power DDR1. These devices have a density of 32 Mbits and are organized as two internal banks. To use this feature, the user has to define the type of memory and the data bus width (see 35.7.8 MPDDRC_MD).<br>The 16-bit memory device is organized as 2 banks, 9 columns and 11 rows.<br>The 32-bit memory device is organized as 2 banks, 8 columns and 11 rows.<br>It is impossible to use two 16-bit memory devices (2 x 32 Mbits) for creating one 32-bit memory device (64 Mbits). In this case, it is recommended to use one 32-bit memory device which embeds four internal banks. |

**Bit 17 – ENRDM** Enable Read Measure
This feature is not supported during a change frequency. See "CHG_FRQ: Change Clock Frequency During Self-refresh Mode".

| Value | Name | Description |
|---|---|---|
| 0 | OFF | DQS/DDR_DATA phase error correction is disabled |
| 1 | ON | DQS/DDR_DATA phase error correction is enabled |

**Bit 16 – DQMS** Mask Data is Shared

| Value | Name | Description |
|---|---|---|
| 0 | NOT_SHARED | DQM is not shared with another controller |
| 1 | SHARED | DQM is shared with another controller |

**Bits 14:12 – OCD[2:0]** Off-chip Driver
SDRAM Controller supports only two values for OCD (default calibration and exit from calibration). These values MUST always be programmed during the initialization sequence. The default calibration must be programmed first, after which the exit calibration and maintain settings must be programmed.

This field is found only in the DDR2-SDRAM devices.

| Value | Name | Description |
|---|---|---|
| 0 | DDR2_EXITCALIB | Exit from OCD Calibration mode and maintain settings |
| 7 | DDR2_DEFAULT_CALIB | OCD calibration default |

**Bits 11:10 – ZQ[1:0]** ZQ Calibration
This parameter is used to calibrate DRAM On resistance (Ron) values over PVT.

This field is found only in the low-power DDR2-SDRAM devices and low-power DDR3-SDRAM devices.

| Value | Name | Description |
|---|---|---|
| 0 | INIT | Calibration command after initialization |
| 1 | LONG | Long calibration |
| 2 | SHORT | Short calibration |
| 3 | RESET | ZQ Reset |

**Bit 9 – DIS_DLL** DISABLE DLL
This value is used during the powerup sequence. It is only found in the DDR2-SDRAM devices and DDR3-SDRAM devices and low-power DDR3-SDRAM devices.

| Value | Description |
|---|---|
| 0 | Enable DLL. |
| 1 | Disable DLL. |

**Bit 8 – DIC_DS** Output Driver Impedance Control (Drive Strength)
This bit name is described as “DS” in some memory datasheets. It defines the output drive strength. This value is used during the powerup sequence.

For DDR3-SDRAM devices, this field is equivalent to ODS, Output Drive Strength.

This bit is found only in the DDR2-SDRAM devices and DDR3-SDRAM devices.

| Value | Name | Description |
|---|---|---|
| 0 | DDR2_NORMALSTRENGTH_DDR3_RZQ_6 | Normal drive strength (DDR2) - RZQ_6 (40 [NOM], DDR3) |
| 1 | DDR2_WEAKSTRENGTH_DDR3_RZQ_7 | Weak drive strength (DDR2) - RZQ_7 (34 [NOM], DDR3) |

**Bit 7 – DLL** Reset DLL
This bit defines the value of Reset DLL.

This value is used during the powerup sequence.

This bit is found only in the DDR2-SDRAM devices and DDR3-SDRAM devices.

| Value | Name | Description |
|---|---|---|
| 0 | RESET_DISABLED | Disable DLL reset |
| 1 | RESET_ENABLED | Enable DLL reset |

**Bits 6:4 – CAS[2:0]** CAS Latency
In the case of DDR3-SDRAM devices, the CAS field must be set to 5 and the SHIFT_SAMPLING field must be set to 2. See “SHIFT_SAMPLING: Shift Sampling Point of Data”. This field is not used to set the DDR3-SDRAM. In the case of DDR3-SDRAM devices, the DLL Off mode sets the CAS Read Latency (CRL) and the CAS Write Latency (CWL) to 6. The latency is automatically set by the controller.

| Value | Name | Description |
|---|---|---|
| 2 | DDR_CAS2 | LPDDR1 CAS Latency 2 |
| 3 | DDR_CAS3 | LPDDR3/DDR2/LPDDR2/LPDDR1 CAS Latency 3 |
| 5 | DDR_CAS5 | DDR3 CAS Latency 5 |
| 6 | DDR_CAS6 | DDR3/LPDDR3 CAS Latency 6 |

© 2018 Microchip Technology Inc.

**Bits 3:2 – NR[1:0]** Number of Row Bits

| Value | Name | Description |
|---|---|---|
| 0 | 11_ROW_BITS | 11 bits to define the row number, up to 2048 rows |
| 1 | 12_ROW_BITS | 12 bits to define the row number, up to 4096 rows |
| 2 | 13_ROW_BITS | 13 bits to define the row number, up to 8192 rows |
| 3 | 14_ROW_BITS | 14 bits to define the row number, up to 16384 rows |

**Bits 1:0 – NC[1:0]** Number of Column Bits

| Value | Name | Description |
|---|---|---|
| 0 | DDR9_MDDR8_COL_BITS | 9 bits to define the column number, up to 512 columns, for DDR2/DDR3/LPDDR2/LPDDR3-SDRAM<br>8 bits to define the column number, up to 256 columns, for LPDDR1-SDRAM |
| 1 | DDR10_MDDR9_COL_BITS | 10 bits to define the column number, up to 1024 columns, for DDR2/DDR3/LPDDR2/LPDDR3-SDRAM<br>9 bits to define the column number, up to 512 columns, for LPDDR1-SDRAM |
| 2 | DDR11_MDDR10_COL_BITS | 11 bits to define the column number, up to 2048 columns, for DDR2/DDR3/LPDDR2/LPDDR3-SDRAM<br>SDRAM<br>10 bits to define the column number, up to 1024 columns, for LPDDR1-SDRAM |
| 3 | DDR12_MDDR11_COL_BITS | 12 bits to define the column number, up to 4096 columns, for DDR2/DDR3/LPDDR2/LPDDR3-SDRAM<br>11 bits to define the column number, up to 2048 columns, for LPDDR1-SDRAM |

### 35.7.4 MPDDRC Timing Parameter 0 Register

**Name:** MPDDRC_TPR0
**Offset:** 0x0C
**Reset:** 0x20227225
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TMRD[3:0] | | | | | TWTR[2:0] | | |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 1 | 0 | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TRRD[3:0] | | | | TRP[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TRC[3:0] | | | | TWR[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TRCD[3:0] | | | | TRAS[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |

**Bits 31:28 – TMRD[3:0]** Load Mode Register Command to Activate or Refresh Command
This field defines the delay between a Load mode register command and an Activate or Refresh command in number of DDRCK clock cycles. The number of cycles is between 0 and 15. For low-power DDR2-SDRAM and low-power DDR3-SDRAM, this field is equivalent to $t_{MRW}$.

**Bits 26:24 – TWTR[2:0]** Internal Write to Read Delay
This field defines the internal Write to Read command time in number of DDRCK clock cycles. The number of cycles is between 1 and 7.

**Bits 23:20 – TRRD[3:0]** Active BankA to Active BankB
This field defines the delay between an Activate command in BankA and an Activate command in BankB in number of DDRCK clock cycles. The number of cycles is between 1 and 15.

**Bits 19:16 – TRP[3:0]** Row Precharge Delay
This field defines the delay between a Precharge command and another command in number of DDRCK clock cycles. The number of cycles is between 0 and 15.

**Bits 15:12 – TRC[3:0]** Row Cycle Delay
This field defines the delay between an Activate command and a Refresh command in number of DDRCK clock cycles. The number of cycles is between 0 and 15.

**Bits 11:8 – TWR[3:0]** Write Recovery Delay
This field defines the Write Recovery Time in number of DDRCK clock cycles. The number of cycles is between 1 and 15.

**Bits 7:4 – TRCD[3:0]** Row to Column Delay
This field defines the delay between an Activate command and a Read/Write command in number of DDRCK clock cycles. The number of cycles is between 0 and 15.

**Bits 3:0 – TRAS[3:0]** Active to Precharge Delay
This field defines the delay between an Activate command and a Precharge command in number of DDRCK clock cycles. The number of cycles is between 0 and 15.

### 35.7.5 MPDDRC Timing Parameter 1 Register

**Name:** MPDDRC_TPR1
**Offset:** 0x10
**Reset:** 0x03C80808
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TXP[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TXSRD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TXSNR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TRFC[6:0] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

**Bits 27:24 – TXP[3:0]** Exit Powerdown Delay to First Command
This field defines the delay between CKE set high and a valid command in number of DDRCK clock cycles. The number of cycles is between 0 and 15.

**Bits 23:16 – TXSRD[7:0]** Exit Self-refresh Delay to Read Command
This field defines the delay between CKE set high and a Read command in number of DDRCK clock cycles. The number of cycles is between 0 and 255.

This field is found only in DDR2-SDRAM and DDR3-SDRAM devices.

In case of DDR3-SDRAM, this field is equivalent to $t_{XSDLL}$. In DLL Off mode, this timing is not used. The field must be set to 0.

**Bits 15:8 – TXSNR[7:0]** Exit Self-refresh Delay to Non-Read Command
This field defines the delay between CKE set high and a Non Read command in number of DDRCK clock cycles. The number of cycles is between 0 and 255. This field is used by the DDR-SDRAM devices. In case of low-power DDR-SDRAM, this field is equivalent to $t_{XSR}$. In case of DDR3-SDRAM, this field is equivalent to $t_{XS}$.

**Bits 6:0 – TRFC[6:0]** Row Cycle Delay
This field defines the delay between a Refresh command or a Refresh and Activate command in number of DDRCK clock cycles. The number of cycles is between 0 and 127.

In case of low-power DDR2-SDRAM and low-power DDR3-SDRAM, this field is equivalent to $t_{RFCab}$. If the user enables the function “Refresh Per Bank” (see “REF_PB: Refresh Per Bank”), this field is equivalent to $t_{RFCpb}$.

### 35.7.6 MPDDRC Timing Parameter 2 Register

**Name:** MPDDRC_TPR2
**Offset:** 0x14
**Reset:** 0x00042062
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TFAW[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 1 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | TRTP[2:0] | | | TRPA[3:0] | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXARDS[3:0] | | | | TXARD[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 |

**Bits 19:16 – TFAW[3:0]** Four Active Windows
DDR2 and DDR3 devices with eight banks (1 Gbit or larger) have an additional requirement concerning $t_{FAW}$ timing. This requires that no more than four Activate commands may be issued in any given $t_{FAW}$ (MIN) period.

The number of cycles is between 0 and 15.

This field is found only in DDR2-SDRAM and LPDDR2-SDRAM and DDR3-SDRAM and LPDDR3-SDRAM devices.

**Bits 14:12 – TRTP[2:0]** Read to Precharge
This field defines the delay between a Read command and a Precharge command in number of DDRCK clock cycles.

The number of cycles is between 0 and 7.

**Bits 11:8 – TRPA[3:0]** Row Precharge All Delay
This field defines the delay between a Precharge All Banks command and another command in number of DDRCK clock cycles. The number of cycles is between 0 and 15.

This field is found only in the DDR2-SDRAM devices.

In the case of LPDDR2-SDRAM and LPDDR3-SDRAM, this field is equivalent to $t_{RPAB}$.

**Bits 7:4 – TXARDS[3:0]** Exit Active Power Down Delay to Read Command in Mode “Slow Exit”
This field defines the delay between CKE set high and a Read command in number of DDRCK clock cycles. The number of cycles is between 0 and 15.

This field is found only in the DDR2-SDRAM devices.

**Bits 3:0 – TXARD[3:0]** Exit Active Power Down Delay to Read Command in Mode “Fast Exit”
This field defines the delay between CKE set high and a Read command in number of DDRCK clock cycles. The number of cycles is between 0 and 15.

This field is found only in the DDR2-SDRAM devices.

### 35.7.7 MPDDRC Low-Power Register

**Name:** MPDDRC_LPR
**Offset:** 0x1C
**Reset:** 0x00010000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | SELF_DONE | CHG_FRQ |
| Access | | | | | | | R | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | UPD_MR[1:0] | | | | | APDE |
| Access | | | R/W | R/W | | | | R/W |
| Reset | | | 0 | 0 | | | | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | TIMEOUT[1:0] | | | DS[2:0] | | |
| Access | | | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | 0 | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | PASR[2:0] | | | LPDDR2_LPDDR3_PWOFF | CLK_FR | LPCB[1:0] | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 25 – SELF_DONE** Self-refresh is Done
This bit indicates that external device is in Self-refresh mode.

**Bit 24 – CHG_FRQ** Change Clock Frequency During Self-refresh Mode
This mode allows to change the Low-power DDR-DRAM input clock frequency. This mode is unique to the Low-power DDR-DRAM devices.

**Bits 21:20 – UPD_MR[1:0]** Update Load Mode Register and Extended Mode Register
This bit is used to enable or disable automatic update of the Load Mode Register and Extended Mode Register. This update depends on the MPDDRC integration in a system. MPDDRC can either share or not an external bus with another controller.

| Value | Name | Description |
|---|---|---|
| 0 | NO_UPDATE | Update of Load Mode and Extended Mode registers is disabled. |
| 1 | UPDATE_SHAREDBUS | MPDDRC shares an external bus. Automatic update is done during a refresh command and a pending read or write access in the SDRAM device. |
| 2 | UPDATE_NOSHAREDBUS | MPDDRC does not share an external bus. Automatic update is done before entering Self-refresh mode. |

| Value | Name | Description |
|---|---|---|
| 3 | – | Reserved |

**Bit 16 – APDE** Active Power Down Exit Time
This mode is unique to the DDR2-SDRAM and DDR3-SDRAM devices.

This mode manages the active Powerdown mode which determines performance versus power saving.

After the initialization sequence, as soon as the APDE field is modified, the Extended Mode Register (located in the memory of the external device) is accessed automatically and APDE bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access

| Value | Name | Description |
|---|---|---|
| 0 | DDR2_FAST_EXIT | Fast Exit from Power Down. DDR2-SDRAM and DDR3-SDRAM devices only. |
| 1 | DDR2_SLOW_EXIT | Slow Exit from Power Down. DDR2-SDRAM and DDR3-SDRAM devices only. |

**Bits 13:12 – TIMEOUT[1:0]** Time Between Last Transfer and Low-Power Mode
This field defines when Low-power mode is activated.

| Value | Name | Description |
|---|---|---|
| 0 | NONE | SDRAM Low-power mode is activated immediately after the end of the last transfer. |
| 1 | DELAY_64_CLK | SDRAM Low-power mode is activated 64 clock cycles after the end of the last transfer. |
| 2 | DELAY_128_CLK | SDRAM Low-power mode is activated 128 clock cycles after the end of the last transfer. |
| 3 | – | Reserved |

**Bits 10:8 – DS[2:0]** Drive Strength
This field is unique to low-power DDR1-SDRAM. It selects the output drive strength.

After the initialization sequence, as soon as the DS field is modified, the Extended Mode Register is accessed automatically and DS bits are updated. Depending on the UPD_MR bit, update is done before entering self-refresh mode or during a refresh command and a pending read or write access.

| Value | Name | Description |
|---|---|---|
| 0 | DS_FULL | Full drive strength |
| 1 | DS_HALF | Half drive strength |
| 2 | DS_QUARTER | Quarter drive strength |
| 3 | DS_OCTANT | Octant drive strength |
| 4–7 | – | Reserved |

**Bits 6:4 – PASR[2:0]** Partial Array Self-refresh
This field is unique to low-power DDR1-SDRAM. It is used to specify whether only one-quarter, one-half or all banks of the DDR-SDRAM array are enabled. Disabled banks are not refreshed in Self-refresh mode.

The values of this field are dependent on the low-power DDR-SDRAM devices.

After the initialization sequence, as soon as the PASR field is modified, the Extended Mode Register in the external device memory is accessed automatically and PASR bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access.

**Bit 3 – LPDDR2_LPDDR3_PWOFF** LPDDR2/3 Power Off Bit
LPDDR2/3 power off sequence must be controlled to preserve the LPDDR2/3 device. The power failure is handled at system level (IRQ or FIQ) and the LPDDR2/3 power off sequence is applied using the LPDDR2_LPDDR3_PWOFF bit.

LPDDR2_LPDDR3_PWOFF bit is used to impose CKE low before a power off sequence. Uncontrolled power off sequence can be applied only up to 400 times in the life of a LPDDR2/3 device.

| Value | Name | Description |
|---|---|---|
| 0 | DISABLED | No power-off sequence applied to LPDDR2/3. |
| 1 | ENABLED | A power-off sequence is applied to the LPDDR2/3 device. CKE is forced low. |

**Bit 2 – CLK_FR** Clock Frozen Command Bit
This field sets the clock low during Powerdown mode. Some DDR-SDRAM devices do not support freezing the clock during Powerdown mode. Refer to the relevant DDR-SDRAM device datasheet for details.

| Value | Name | Description |
|---|---|---|
| 0 | DISABLED | Clock(s) is/are not frozen. |
| 1 | ENABLED | Clock(s) is/are frozen. |

**Bits 1:0 – LPCB[1:0]** Low-power Command Bit

| Value | Name | Description |
|---|---|---|
| 0 | NOLOWPOWER | Low-power feature is inhibited. No Powerdown, Self-refresh and Deep power modes are issued to the DDR-SDRAM device. |
| 1 | SELFREFRESH | The MPDDRC issues a self-refresh command to the DDR-SDRAM device, the clock(s) is/are deactivated and the CKE signal is set low. The DDR-SDRAM device leaves the Self-refresh mode when accessed and reenters it after the access. |
| 2 | POWERDOWN | The MPDDRC issues a Powerdown command to the DDR-SDRAM device after each access, the CKE signal is set low. The DDR-SDRAM device leaves the Powerdown mode when accessed and reenters it after the access. |
| 3 | DEEPPOWERDOWN | The MPDDRC issues a Deep Powerdown command to the low-power DDR-SDRAM device. |

### 35.7.8 MPDDRC Memory Device Register

**Name:** MPDDRC_MD
**Offset:** 0x20
**Reset:** 0x00000013
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IO_WIDTH[1:0] | | DENSITY[3:0] | | | | TYPE[1:0] | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | REV_ID[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MANU_ID[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RL3 | WL | | DBW | | MD[2:0] | | |
| Access | R | R | | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | | 1 | | 0 | 1 | 1 |

**Bits 31:30 – IO_WIDTH[1:0]** Width of Memory
This field gives the width of the memory. This field is unique to low-power DDR2-SDRAM and low-power DDR3-SDRAM. When Backup mode is used, this information is lost after Backup mode exit.

| Value | Name | Description |
|---|---|---|
| 0 | WIDTH_32 | The data bus width is 32 bits. |
| 1 | WIDTH_16 | The data bus width is 16 bits. |
| 2 | WIDTH_8 | The data bus width is 8 bits. |
| 3 | NOT_USED | – |

**Bits 29:26 – DENSITY[3:0]** Density of Memory
This field is unique to low-power DDR2-SDRAM and low-power DDR3-SDRAM. When Backup mode is used, this information is lost after Backup mode exit

This field gives the density of the memory.

| Value | Name | Description |
|---|---|---|
| 0 | DENSITY_64MBITS | The device density is 64 Mbits. |
| 1 | DENSITY_128MBITS | The device density is 128 Mbits. |
| 2 | DENSITY_256MBITS | The device density is 256 Mbits. |

| Value | Name | Description |
|---|---|---|
| 3 | DENSITY_512MBITS | The device density is 512 Mbits. |
| 4 | DENSITY_1GBITS | The device density is 1 Gbit. |
| 5 | DENSITY_2GBITS | The device density is 2 Gbits. |
| 6 | DENSITY_4GBITS | The device density is 4 Gbits. |
| 7 | DENSITY_8GBITS | The device density is 8 Gbits. |
| 8 | DENSITY_16GBITS | The device density is 16 Gbits. |
| 9 | DENSITY_32GBITS | The device density is 32 Gbits. |

**Bits 25:24 – TYPE[1:0]** DRAM Architecture
This field gives the DRAM architecture. This field is unique to low-power DDR2-SDRAM and low-power DDR3-SDRAM. When Backup mode is used, this information is lost after Backup mode exit.

| Value | Name | Description |
|---|---|---|
| 0 | S4_SDRAM | 4n prefetch architecture |
| 1 | S2_SDRAM | 2n prefetch architecture |
| 2 | NVM | Non-volatile device |
| 3 | S8_SDRAM | 8n prefetch architecture |

**Bits 23:16 – REV_ID[7:0]** Revision Identification
This field gives the revision ID. This field is unique to low-power DDR2-SDRAM and low-power DDR3-SDRAM. When Backup mode is used, this information is lost after Backup mode exit.

**Bits 15:8 – MANU_ID[7:0]** Manufacturer Identification
This field gives information concerning the Manufacturer ID. For more information concerning the Manufacturer ID, Refer to document JC-42.6 “Manufacturer Identification (ID) Code for Low Power Memories”. This field is unique to low-power DDR2-SDRAM and low-power DDR3-SDRAM. When Backup mode is used, this information is lost after Backup mode exit.

**Bit 7 – RL3** Read Latency 3 Option Support
This field gives information concerning the read latency supported. Read latency 3 has been defined per Jedec for frequency ≤ 166 MHz. This feature is optional. If the LPDDR3 device does not support this feature, a CAS latency of 6 is used. This field is unique to low-power DDR3-SDRAM. When Backup mode is used, this information is lost after Backup mode exit.

| Value | Name | Description |
|---|---|---|
| 0 | RL3_SUPPORT | Read latency of 3 is supported |
| 1 | RL3_NOT_SUPPORTED | Read latency of 3 is not supported |

**Bit 6 – WL** Write Latency
This field gives the write latency supported by the memory device. This field is unique to low-power DDR3-SDRAM. When Backup mode is used, this information is lost after Backup mode exit.

| Value | Name | Description |
|---|---|---|
| 0 | WL_SETA | Write Latency Set A |
| 1 | WL_SETB | Write Latency Set B |

**Bit 4 – DBW** Data Bus Width

| Value | Name | Description |
|---|---|---|
| 0 | DBW_32_BITS | Data bus width is 32 bits |

| Value | Name | Description |
|---|---|---|
| 1 | DBW_16_BITS | Data bus width is 16 bits. |

**Bits 2:0 – MD[2:0]** Memory Device

| Value | Name | Description |
|---|---|---|
| 3 | LPDDR_SDRAM | Low-power DDR1-SDRAM |
| 4 | DDR3_SDRAM | DDR3-SDRAM |
| 5 | LPDDR3_SDRAM | Low-power DDR3-SDRAM |
| 6 | DDR2_SDRAM | DDR2-SDRAM |
| 7 | LPDDR2_SDRAM | Low-power DDR2-SDRAM |

### 35.7.9 MPDDRC Low-power DDR2 Low-power DDR3 Low-power Register

**Name:** MPDDRC_LPDDR23_LPR
**Offset:** 0x28
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | DS[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SEG_MASK[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SEG_MASK[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BK_MASK_PASR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27:24 – DS[3:0]** Drive Strength
After the initialization sequence, as soon as the DS field is modified, Mode Register 3 is accessed automatically and DS bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access.

This field is unique to low-power DDR2-SDRAM and low-power DDR3-SDRAM. It selects the I/O drive strength as shown in the table below.

In case of low-power DDR2-SDRAM or low-power DDR3-SDRAM, the RDIV field in the MPDDRC_IO_CALIBR register must be set to same value of DS field.

| Value | Name | Description |
|---|---|---|
| 0 | – | Reserved |
| 1 | DS_34_3 | 34.3 ohm typical |
| 2 | DS_40 | 40 ohm typical (default) |
| 3 | DS_48 | 48 ohm typical |
| 4 | DS_60 | 60 ohm typical |
| 5 | – | Reserved |
| 6 | DS_80 | 80 ohm typical |
| 7 | DS_120 | 120 ohm typical |
| 8–15 | – | Reserved |

**Bits 23:8 – SEG_MASK[15:0]** Segment Mask Bit
After the initialization sequence, as soon as the SEG_MASK field is modified, Mode Register 17 is accessed automatically and SEG_MASK bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access.

This mode is unique to the low-power DDR2-SDRAM-S4 and low-power DDR3-SDRAM devices. The number of Segment Mask bits differs with the density. For 1 Gbit density, 8 segments are used. In Self-refresh mode, when the Segment Mask bit is configured, the refresh operation is masked in the segment.

| Value | Description |
|---|---|
| 0 | Segment is refreshed (= unmasked). |
| 1 | Segment is not refreshed (= masked). |

**Bits 7:0 – BK_MASK_PASR[7:0]** Bank Mask Bit/PASR
Partial Array Self-Refresh (low-power DDR2-SDRAM-S4 devices and low-power DDR3-SDRAM only)

After the initialization sequence, as soon as the BK_MASK_PASR field is modified, Mode Register 16 is accessed automatically and BK_MASK_PASR bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access.

This mode is unique to the low-power DDR2-SDRAM-S4 and low-power DDR3-SDRAM devices. In Self-refresh mode, each bank of LPDDR2/LPDDR3 can be independently configured whether a self-refresh operation is taking place or not.

After the initialization sequence, as soon as the BK_MASK_PASR field is modified, the Extended Mode Register is accessed automatically and BK_MASK_PASR bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access.

| Value | Description |
|---|---|
| 0 | Refresh is enabled (= unmasked). |
| 1 | Refresh is disabled (= masked). |

### 35.7.10 MPDDRC Low-power DDR2 Low-power DDR3 and DDR3 Calibration and MR4 Register

**Name:** MPDDRC_LPDDR2_LPDDR3_DDR3_CAL_MR4
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | MR4_READ[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MR4_READ[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COUNT_CAL[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COUNT_CAL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – MR4_READ[15:0]** Mode Register 4 Read Interval
MR4_READ defines the time period between MR4 reads (for LPDDR2-SDRAM). The formula is (MR4_READ+1) × $t_{REF}$. The value to be loaded depends on the average time between REFRESH commands, $t_{REF}$. For example, for an LPDDR2-SDRAM with the time between refresh of 7.8 µs, if the MR4_READ value is 2, the time period between MR4 reads is 23.4 µs.

The LPDDR2-SDRAM and LPDDR3-SDRAM devices feature a temperature sensor whose status can be read from MR4 register. This sensor can be used to determine an appropriate refresh rate. Temperature sensor data may be read from MR4 register using the Mode Register Read protocol. The Adjust Refresh Rate bit (ADJ_REF) in the Refresh Timer Register (MPDDRC_RTR) must be written to a one to activate these reads.

**Bits 15:0 – COUNT_CAL[15:0]** LPDDR2 LPDDR3 and DDR3 Calibration Timer Count
This 16-bit field is loaded into a timer which generates the calibration pulse. Each time the calibration pulse is generated, a ZQCS calibration sequence is initiated.

The ZQCS Calibration command is used to calibrate DRAM Ron values over PVT.

One ZQCS command can effectively correct at least 1.5% of output impedance errors within $T_{zqcs}$.

One method for calculating the interval between ZQCS commands gives the temperature ($T_{driftrate}$) and voltage ($V_{driftrate}$) drift rates to which the SDRAM is subject in the application. The interval could be defined by the following formula:

ZQCorrection/(($T_{Sens}$ x $T_{driftrate}$) + ($V_{Sens}$ x $V_{driftrate}$))

where $T_{Sens}$ = max(dRONdTM) and $V_{Sens}$ = max(dRONdVM) define the SDRAM temperature and voltage sensitivities.

For example, if $T_{Sens}$ = 0.75%/C, $V_{Sens}$ = 0.2%/mV, $T_{driftrate}$ = 1C/sec and $V_{driftrate}$ = 15 mV/s, then the interval between ZQCS commands is calculated as:

1.5/((0.75 x 1) + (0.2 x 15)) = 0.4s

In this example, the devices require a calibration every 0.4s. The value to be loaded depends on average time between REFRESH commands, $t_{REF}$.

For example, for a device with the time between refresh of 7.8 µs, the value of the Calibration Timer Count field is programmed: (0.4/7.8 x $10^{-6}$) = 0xC852.

$T_{Sens}$ and $V_{Sens}$ are given by the manufacturer (Output Driver Sensitivity definition). $T_{driftrate}$ and $V_{driftrate}$ are defined by the end user.

### 35.7.11 MPDDRC Low-power DDR2 Low-power DDR3 and DDR3 Timing Calibration Register

**Name:** MPDDRC_LPDDR2_LPDDR3_DDR3_TIM_CAL
**Offset:** 0x30
**Reset:** 0x00000006
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RZQI[1:0] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ZQCS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |

**Bits 17:16 – RZQI[1:0]** Built-in Self-Test for RZQ Information
This field indicates whether the device has detected a resistor connection to the ZQ pin.

This mode is unique to low-power DDR3-SDRAM devices.

| Value | Name | Description |
|---|---|---|
| 0 | RZQ_NOT_SUPPORTED | RZQ self test not supported |
| 1 | ZQ_VDDCA_FLOAT | The ZQ pin can be connected to VDDCA or left floating. |
| 2 | ZQ_SHORTED_GROUND | The ZQ pin can be shorted to ground. |
| 3 | ZQ_SELF_TEST_OK | ZQ pin self test complete; no error condition detected |

**Bits 7:0 – ZQCS[7:0]** ZQ Calibration Short
This field defines the delay between the ZQ Calibration command and any valid command in number of DDRCK clock cycles.

The number of cycles is between 0 and 255. This field applies to LPDDR2, LPDDR3 and DDR3 devices.

### 35.7.12 MPDDRC I/O Calibration Register

**Name:** MPDDRC_IO_CALIBR
**Offset:** 0x34
**Reset:** 0x00870000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CALCODEN[3:0] | | | | CALCODEP[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | TZQIO[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | EN_CALIB | | RDIV[2:0] | | |
| Access | | | | R/W | | R/W | R/W | R/W |
| Reset | | | | 0 | | 0 | 0 | 0 |

**Bits 23:20 – CALCODEN[3:0]** Number of N-type Transistors
This value gives the number of N-type transistors to perform the calibration.

**Bits 19:16 – CALCODEP[3:0]** Number of P-type Transistors
This value gives the number of P-type transistors to perform the calibration.

**Bits 14:8 – TZQIO[6:0]** IO Calibration
This field defines the delay between the start up of the amplifier and the beginning of the calibration in number of DDRCK clock cycles. The value of this field must be set to 600 ns.

The number of cycles is between 0 and 127.

The TZQIO configuration code must be set correctly depending on the clock frequency using the following formula:

TZQIO = (DDRCK × 600e-9) + 1

where DDRCK frequency is in Hz.

For example, for a frequency of 176 MHz, the value of the TZQIO field is configured (176 × 10e6) × (600e-9) + 1.

**Bit 4 – EN_CALIB** Enable Calibration
This field enables calibration for the LPDDR1 and DDR2 devices. When the calibration is enabled, it is recommended to define the COUNT_CAL field (see "COUNT_CAL: LPDDR2 LPDDR3 and DDR3 Calibration Timer Count").

This 16-bit field is loaded into a timer which generates the calibration pulse. Each time the calibration pulse is generated, a calibration sequence is initiated.

| Value | Name | Description |
|---|---|---|
| 0 | DISABLE_CALIBRATION | Calibration is disabled. |
| 1 | ENABLE_CALIBRATION | Calibration is enabled. |

**Bits 2:0 – RDIV[2:0]** Resistor Divider, Output Driver Impedance
RDIV is used with the external precision resistor RZQ to define the output driver impedance. The value of RZQ is either 24K ohms (LPDDR2/LPDDR3 device) or 23K ohms (DDR3L device) or 22K ohms (DDR3 device) or 21K ohms (DDR2/LPDDR1 device).

| Value | Name | Description |
|---|---|---|
| 0 | Reserved | Reserved |
| 2 | RZQ_40_RZQ_38_RZQ_37_RZQ_35 | LPDDR2 serial impedance line = 40 ohms,<br>LPDDR3 serial impedance line = 38 ohms,<br>DDR3 serial impedance line = 37 ohms,<br>DDR2/LPDDR1 serial impedance line = 35 ohms |
| 3 | RZQ_48_RZQ_46_RZQ_44_RZQ_43 | LPDDR2 serial impedance line = 48 ohms,<br>LPDDR3 serial impedance line = 46 ohms,<br>DDR3 serial impedance line = 44 ohms,<br>DDR2/LPDDR1 serial impedance line = 43 ohms |
| 4 | RZQ_60_RZQ_57_RZQ_55_RZQ_52 | LPDDR2 serial impedance line = 60 ohms,<br>LPDDR3 serial impedance line = 57 ohms,<br>DDR3 serial impedance line = 55 ohms,<br>DDR2/LPDDR1 serial impedance line = 52 ohms |
| 6 | RZQ_80_RZQ_77_RZQ_73_RZQ_70 | LPDDR2 serial impedance line = 80 ohms,<br>LPDDR3 serial impedance line = 77 ohms,<br>DDR3 serial impedance line = 73 ohms,<br>DDR2/LPDDR1 serial impedance line = 70 ohms |
| 7 | RZQ_120_RZQ_115_RZQ_110_RZQ_105 | LPDDR2 serial impedance line = 120 ohms,<br>LPDDR3 serial impedance line = 115 ohms,<br>DDR3 serial impedance line = 110 ohms,<br>DDR2/LPDDR1 serial impedance line = 105 ohms |

### 35.7.13 MPDDRC OCMS Register

**Name:** MPDDRC_OCMS
**Offset:** 0x38
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SCR_EN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – SCR_EN** Scrambling Enable

| Value | Description |
|---|---|
| 0 | Disables “Off-chip” scrambling for SDRAM access. |
| 1 | Enables “Off-chip” scrambling for SDRAM access. |

### 35.7.14 MPDDRC OCMS KEY1 Register

**Name:** MPDDRC_OCMS_KEY1
**Offset:** 0x3C
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register. The register can only be written once.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY1[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEY1[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | KEY1[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | KEY1[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 31:0 – KEY1[31:0]** Off-chip Memory Scrambling (OCMS) Key Part 1
When Off-chip Memory Scrambling is enabled, the data scrambling depends on KEY1 and KEY2 values.

### 35.7.15 MPDDRC OCMS KEY2 Register

**Name:** MPDDRC_OCMS_KEY2
**Offset:** 0x40
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register. The register can only be written once.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY2[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEY2[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | KEY2[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | KEY2[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 31:0 – KEY2[31:0]** Off-chip Memory Scrambling (OCMS) Key Part 2
When Off-chip Memory Scrambling is enabled, the data scrambling depends on KEY1 and KEY2 values.

### 35.7.16 MPDDRC Configuration Arbiter Register

**Name:** MPDDRC_CONF_ARBITER
**Offset:** 0x44
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BDW_BURST_P7 | BDW_BURST_P6 | BDW_BURST_P5 | BDW_BURST_P4 | BDW_BURST_P3 | BDW_BURST_P2 | BDW_BURST_P1 | BDW_BURST_P0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MA_PR_P7 | MA_PR_P6 | MA_PR_P5 | MA_PR_P4 | MA_PR_P3 | MA_PR_P2 | MA_PR_P1 | MA_PR_P0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RQ_WD_P7 | RQ_WD_P6 | RQ_WD_P5 | RQ_WD_P4 | RQ_WD_P3 | RQ_WD_P2 | RQ_WD_P1 | RQ_WD_P0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | BDW_MAX_CUR | | ARB[1:0] | |
| Access | | | | | R/W | | R/W | R/W |
| Reset | | | | | 0 | | 0 | 0 |

**Bits 24, 25, 26, 27, 28, 29, 30, 31 – BDW_BURST_Px** Bandwidth Arbitration Mode on Port X

| Value | Description |
|---|---|
| 0 | The arbitration is done when the bandwidth limit defined in MPDDRC_BDW_PORT_0123/4567.BDW_Px is reached. If the bandwidth limit is reached during a burst access, the burst is completed. |
| 1 | The arbitration is done when the bandwidth limit defined in MPDDRC_BDW_PORT_0123/4567.BDW_Px is reached. If the bandwidth limit is reached during a burst access, the burst is broken. |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – MA_PR_Px** Master or Software Provide Information

| Value | Description |
|---|---|
| 0 | Number of requests or words is provided by the master, if the master supports this feature. |
| 1 | Number of requests or words is provided by software, see “NRQ_NWD_BDW_Px: Number of Requests, Number of Words or Bandwidth Allocation from Port 0-1-2-3”. |

**Bits 8, 9, 10, 11, 12, 13, 14, 15 – RQ_WD_Px** Request or Word from Port X

| Value | Description |
|---|---|
| 0 | Number of requests is selected. |
| 1 | Number of words is selected. |

**Bit 3 – BDW_MAX_CUR** Bandwidth Max or Current
This field displays the maximum of the bandwidth or the current bandwidth for each port.

The maximum of the bandwidth is computed when at least two ports of MPDDRC are used.

That information is given in MPDDRC Current/Maximum Bandwidth Port 0-1-2-3 Register and MPDDRC Current/Maximum Bandwidth Port 4-5-6-7 Register.

| Value | Description |
|---|---|
| 0 | Current bandwidth is displayed. |
| 1 | Maximum of the bandwidth is displayed. |

**Bits 1:0 – ARB[1:0]** Type of Arbitration
This field allows to choose the type of arbitration: round-robin, number of requests per port or bandwidth per port.

| Value | Name | Description |
|---|---|---|
| 0 | ROUND | Round-Robin Policy |
| 1 | NB_REQUEST | Request Policy |
| 2 | BANDWIDTH | Bandwidth Policy |
| 3 | – | Reserved |

### 35.7.17 MPDDRC Timeout Register

**Name:** MPDDRC_TIMEOUT
**Offset:** 0x48
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TIMEOUT_P7[3:0] | | | | TIMEOUT_P6[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TIMEOUT_P5[3:0] | | | | TIMEOUT_P4[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TIMEOUT_P3[3:0] | | | | TIMEOUT_P2[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TIMEOUT_P1[3:0] | | | | TIMEOUT_P0[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0:3, 4:7, 8:11, 12:15, 16:19, 20:23, 24:27, 28:31 – TIMEOUT_Px** Timeout for Ports 0, 1, 2, 3, 4, 5, 6, 7
Some masters have the particularity to insert idle state between two accesses. This field defines the delay between two accesses on the same port in number of DDRCK clock cycles before arbitration and handling the access over to another port.

This field is not used with round-robin and bandwidth arbitrations.

The number of cycles is between 1 and 15.

### 35.7.18 MPDDRC Request Port 0-1-2-3 Register

**Name:** MPDDRC_REQ_PORT_0123
**Offset:** 0x4C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NRQ_NWD_BDW_P3[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NRQ_NWD_BDW_P2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NRQ_NWD_BDW_P1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NRQ_NWD_BDW_P0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0:7, 8:15, 16:23, 24:31 – NRQ_NWD_BDW_Px** Number of Requests, Number of Words or Bandwidth Allocation from Port 0-1-2-3
The number of requests corresponds to the number of start transfers. For example, setting this field to 2 performs two burst accesses regardless of the burst type (INCR4, INCR8, etc.). The number of words corresponds exactly to the number of accesses; setting this field to 2 performs two accesses. In this example, burst accesses will be broken.

These values depend on scheme arbitration (see MPDDRC Configuration Arbiter Register).

In case of round-robin arbitration, this field is not used. In case of “bandwidth arbitration”, this field corresponds to percentage allocated for each port. In case of “request” arbitration, this field corresponds to number of start transfers or to number of accesses allocated for each port.

### 35.7.19 MPDDRC Request Port 4-5-6-7 Register

**Name:** MPDDRC_REQ_PORT_4567
**Offset:** 0x50
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NRQ_NWD_BDW_P7[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NRQ_NWD_BDW_P6[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NRQ_NWD_BDW_P5[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NRQ_NWD_BDW_P4[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0:7, 8:15, 16:23, 24:31 – NRQ_NWD_BDW_Px** Number of Requests, Number of Words or Bandwidth allocation from port 4-5-6-7
The number of requests corresponds to the number of start transfers. For example, setting this field to 2 performs two burst accesses regardless of the burst type (INCR4, INCR8,etc.). The number of words corresponds exactly to the number of accesses; setting this field to 2 performs two accesses. In this example, burst accesses will be broken.

These values depend on scheme arbitration (see MPDDRC Configuration Arbiter Register).

In case of round-robin arbitration, this field is not used. In case of “bandwidth arbitration”, this field corresponds to percentage allocated for each port. In case of “request” arbitration, this field corresponds to number of start transfers or to number of accesses allocated for each port.

### 35.7.20 MPDDRC Current/Maximum Bandwidth Port 0-1-2-3 Register

**Name:** MPDDRC_BDW_PORT_0123
**Offset:** 0x54
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | BDW_P3[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | BDW_P2[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | BDW_P1[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | BDW_P0[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0:6, 8:14, 16:22, 24:30 – BDW_Px** Current/Maximum Bandwidth from Port 0-1-2-3
This field displays the current bandwidth or the maximum bandwidth for each port. This information is given in the "BDW_MAX_CUR: Bandwidth Max or Current" field description.

### 35.7.21 MPDDRC Current/Maximum Bandwidth Port 4-5-6-7 Register

**Name:** MPDDRC_BDW_PORT_4567
**Offset:** 0x58
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BDW_P7[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BDW_P6[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

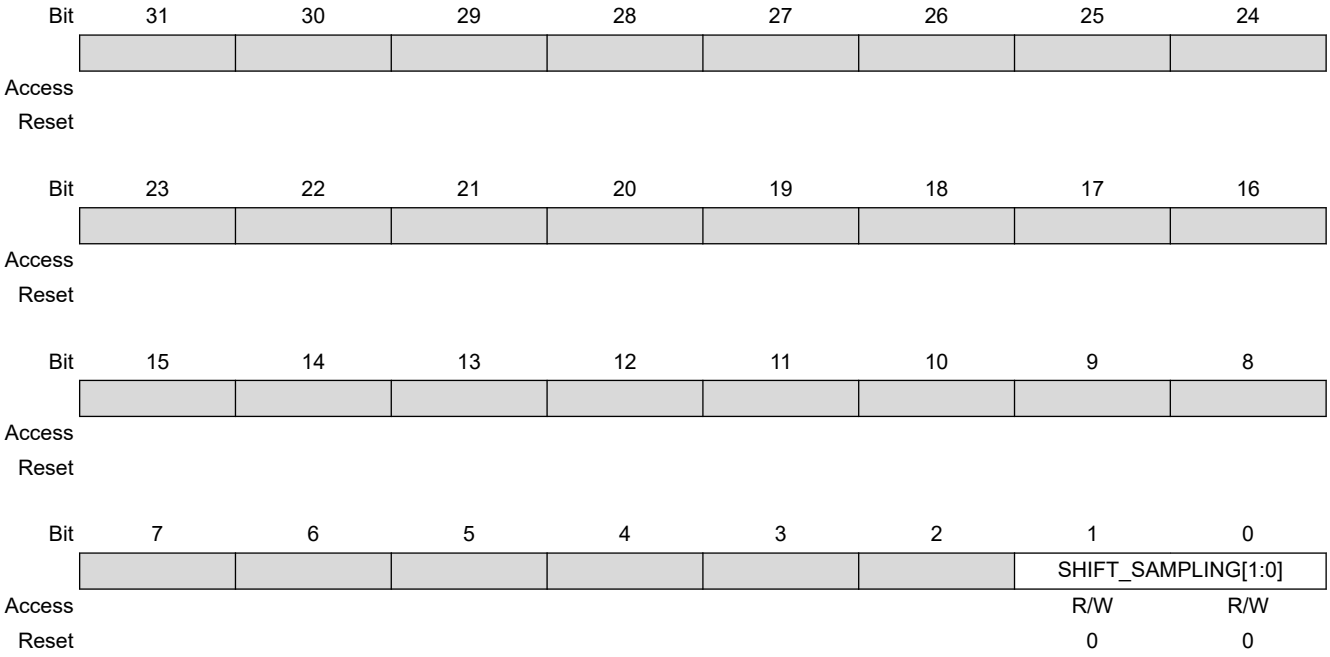
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BDW_P5[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BDW_P4[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0:7, 8:15, 16:23, 24:31 – BDW_Px** Current/Maximum Bandwidth from Port 4-5-6-7
This field displays the current bandwidth or the maximum bandwidth for each port. This information is given in the "BDW_MAX_CUR: Bandwidth Max or Current" field description.

### 35.7.22 MPDDRC Read Data Path Register

**Name:** MPDDRC_RD_DATA_PATH
**Offset:** 0x5C
**Reset:** 0x00000000
**Property:** Read/Write




**Bits 1:0 – SHIFT_SAMPLING[1:0]** Shift Sampling Point of Data
Shifts the sampling point of data coming from the memory device. The higher the memory device clock frequency, the higher the SHIFT_SAMPLING value. Refer to the section "Electrical Characteristics".

| Value | Name | Description |
|---|---|---|
| 0 | NO_SHIFT | Initial sampling point. |
| 1 | SHIFT_ONE_CYCLE | Sampling point is shifted by one cycle. |
| 2 | SHIFT_TWO_CYCLES | Sampling point is shifted by two cycles. |
| 3 | SHIFT_THREE_CYCLES | Sampling point is shifted by three cycles, unique for LPDDR2 and DDR3 and LPDDR3.<br>Not applicable for DDR2 and LPDDR1 devices. |

### 35.7.23 MPDDRC Monitor Configuration Register

**Name:** MPDDRC_MCFGR
**Offset:** 0x60
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | INFO[1:0] | | REFR_CALIB | READ_WRITE[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | RUN | | | SOFT_RESET | EN_MONI |
| Access | | | | R/W | | | R/W | R/W |
| Reset | | | | 0 | | | 0 | 0 |

**Bits 12:11 – INFO[1:0]** Information Type
This field reports information such as latency and the number of transfers monitored on port x [x = 0..7].

| Value | Name | Description |
|---|---|---|
| 0 | MAX_WAIT | Information concerning the transfer with the longest waiting time |
| 1 | NB_TRANSFERS | Number of transfers on the port |
| 2 | TOTAL_LATENCY | Total latency on the port |
| 3 | – | Reserved |

**Bit 10 – REFR_CALIB** Refresh Calibration

| Value | Description |
|---|---|
| 0 | Monitoring does not depend on Autorefresh mode, Self-refresh mode, Powerdown mode, DLL nor calibration impact. |
| 1 | Monitoring depends on Autorefresh mode, Self-refresh mode, Powerdown mode, DLL and calibration impact. |

**Bits 9:8 – READ_WRITE[1:0]** Read/Write Access
This field is used to monitor different types of access.

| Value | Name | Description |
|---|---|---|
| 0 | TRIG_RD_WR | Read and Write accesses are triggered. |

| Value | Name | Description |
|---|---|---|
| 1 | TRIG_WR | Only Write accesses are triggered. |
| 2 | TRIG_RD | Only Read accesses are triggered. |
| 3 | – | Reserved |

**Bit 4 – RUN** Control Monitor

| Value | Description |
|---|---|
| 0 | Monitoring is halted. All counters are stopped. |
| 1 | Monitoring is launched. |

**Bit 1 – SOFT_RESET** Soft Reset

| Value | Description |
|---|---|
| 0 | Soft reset is not performed. |
| 1 | Soft reset is performed. |

**Bit 0 – EN_MONI** Enable Monitor

| Value | Description |
|---|---|
| 0 | Monitor is disabled. |
| 1 | Monitor is enabled. |

### 35.7.24 MPDDRC Monitor Address High/Low Port x Register

**Name:** MPDDRC_MADDRx
**Offset:** 0x64 + x*0x04 [x=0..7]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR_HIGH_PORTx[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR_HIGH_PORTx[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR_LOW_PORTx[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR_LOW_PORTx[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – ADDR_HIGH_PORTx[15:0]** Address High on Port x
Address high which defines the interval to be monitored on port x [x = 0..7]. This address must be programmed according to the memory mapping of the product.

**Bits 15:0 – ADDR_LOW_PORTx[15:0]** Address Low on Port x
Address low which defines the interval to be monitored on port x [x = 0..7]. This address must be programmed according to the memory mapping of the product.

### 35.7.25 MPDDRC Monitor Information Port x Register (MAX_WAIT)

**Name:** MPDDRC_MINFOx (MAX_WAIT)
**Offset:** 0x84 + x*0x04 [x=0..7]
**Reset:** 0x00000000
**Property:** Read-only

The following fields can be read if the INFO field in the MPDDRC Monitor Configuration register is set to 0.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | READ_WRITE |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | SIZE[2:0] | | | | BURST[2:0] | | |
| Access | | R | R | R | | R | R | R |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MAX_PORTx_WAITING[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MAX_PORTx_WAITING[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 24 – READ_WRITE** Read or Write Access on Port x
This field reports the transfer direction for the maximum waiting time.

| Value | Description |
|---|---|
| 0 | Read transfer |
| 1 | Write transfer |

**Bits 22:20 – SIZE[2:0]** Transfer Size on Port x
This field reports the size of the transfer for the maximum waiting time.

| Value | Name | Description |
|---|---|---|
| 0 | 8BITS | Byte transfer |
| 1 | 16BITS | Halfword transfer |
| 2 | 32BITS | Word transfer |
| 3 | 64BITS | Dword transfer |

**Bits 18:16 – BURST[2:0]** Type of Burst on Port x
This field reports the type of burst for the maximum waiting time.

| Value | Name | Description |
|---|---|---|
| 0 | SINGLE | Single transfer |
| 1 | INCR | Incrementing burst of unspecified length |
| 2 | WRAP4 | 4-beat wrapping burst |
| 3 | INCR4 | 4-beat incrementing burst |
| 4 | WRAP8 | 8-beat wrapping burst |
| 5 | INCR8 | 8-beat incrementing burst |
| 6 | WRAP16 | 16-beat wrapping burst |
| 7 | INCR16 | 16-beat incrementing burst |

**Bits 15:0 – MAX_PORTx_WAITING[15:0]** Address High on Port x
This field reports the maximum waiting time and the associated type of transfer (burst, size, read or write).

### 35.7.26 MPDDRC Monitor Information Port x Register (NB_TRANSFERS)

**Name:** MPDDRC_MINFOx (NB_TRANSFERS)
**Offset:** 0x84 + x*0x04 [x=0..7]
**Reset:** 0x00000000
**Property:** Read-only

The following fields can be read if the INFO field in the MPDDRC Monitor Configuration register is set to 1.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | Px_NB_TRANSFERS[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | Px_NB_TRANSFERS[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | Px_NB_TRANSFERS[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | Px_NB_TRANSFERS[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – Px_NB_TRANSFERS[31:0]** Number of Transfers on Port x
This field can be read if the INFO field is set to 1. This field reports the number of transfers performed within an interval (ADDR_HIGH_PORT and ADDR_LOW_PORT) when the port is used.

### 35.7.27 MPDDRC Monitor Information Port x Register (TOTAL_LATENCY)

**Name:** MPDDRC_MINFOx (TOTAL_LATENCY)
**Offset:** 0x84 + x*0x04 [x=0..7]
**Reset:** 0x00000000
**Property:** Read-only

The following fields can be read if the INFO field in the MPDDRC Monitor Configuration register is set to 2.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | Px_TOTAL_LATENCY[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | Px_TOTAL_LATENCY[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | Px_TOTAL_LATENCY[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | Px_TOTAL_LATENCY[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – Px_TOTAL_LATENCY[31:0]** Total Latency on Port x
This field can be read if the INFO field is set to 2. This field reports the total latency within an interval (ADDR_HIGH_PORT and ADDR_LOW_PORT) when the port is used.

### 35.7.28 MPDDRC Write Protection Mode Register

**Name:** MPDDRC_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

See 35.5.7 Register Write Protection for the list of registers that can be protected.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x444452 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit.<br>Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x444452 (“DDR” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x444452 (“DDR” in ASCII). |

### 35.7.29 MPDDRC Write Protection Status Register

**Name:** MPDDRC_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Enable

| Value | Description |
|---|---|
| 0 | No write protection violation occurred since the last read of this register (MPDDRC_WPSR). |
| 1 | A write protection violation occurred since the last read of this register (MPDDRC_WPSR). If this violation is an unauthorized attempt to write a control register, the associated violation is reported into the WPVSRC field. |

# 36. Static Memory Controller (SMC)

## 36.1 Description

This Static Memory Controller (SMC) is capable of handling several types of external memory and peripheral devices, such as SRAM, PSRAM, PROM, EPROM, EEPROM, LCD Module, NOR Flash and NAND Flash.

The SMC generates the signals that control the access to external memory devices or peripheral devices. It has 4 Chip Selects and a 26-bit address bus. The 16-bit data bus can be configured to interface with 8- or 16-bit external devices. Separate read and write control signals allow for direct memory and peripheral interfacing. Read and write signal waveforms are fully parametrizable.

The SMC can manage wait requests from external devices to extend the current access. The SMC is provided with an automatic Slow Clock mode. In Slow Clock mode, it switches from user-programmed waveforms to slow-rate specific waveforms on read and write signals.

The SMC embeds a NAND Flash Controller (NFC). The NFC can handle automatic transfers, sending the commands and address cycles to the NAND Flash and transferring the contents of the page (for read and write) to the NFC SRAM. It minimizes the CPU overhead.

The SMC includes programmable hardware error correcting code with one-bit error correction capability and supports two-bit error detection. In order to improve the overall system performance, the DATA phase of the transfer can be DMA-assisted.

The External Data Bus can be scrambled/unscrambled by means of user keys.

## 36.2 Embedded Characteristics

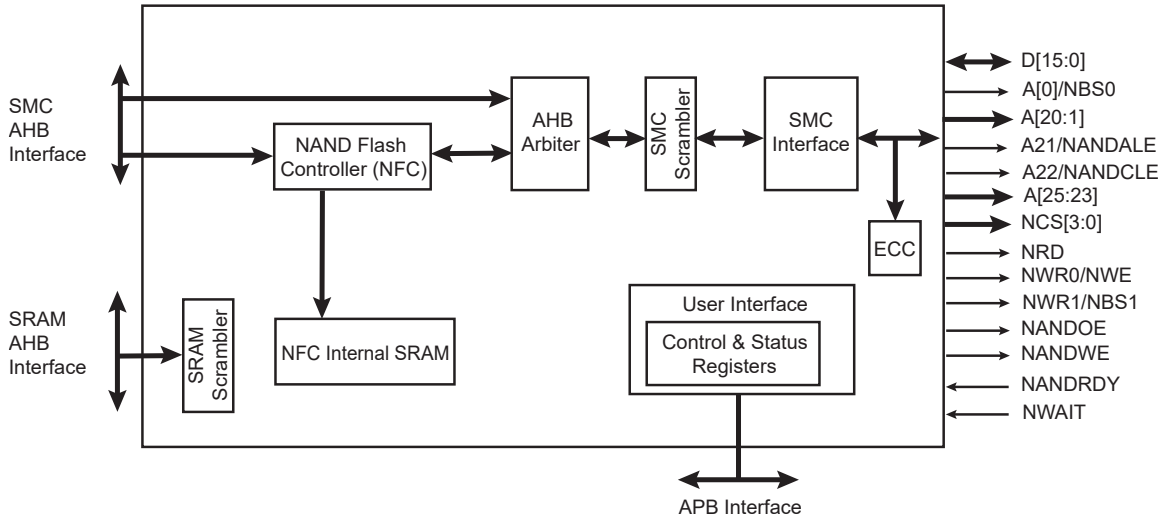
- 64-Mbyte Address Space per Chip Select
- 8- or 16-bit Data Bus
- Word, Halfword, Byte Transfers
- Byte Write or Byte Select Lines
- Programmable Setup, Pulse and Hold Time for Read Signals per Chip Select
- Programmable Setup, Pulse and Hold Time for Write Signals per Chip Select
- Programmable Data Float Time per Chip Select
- External Data Bus Scrambling/Unscrambling Function
- External Wait Request
- Automatic Switch to Slow Clock Mode
- Hardware Configurable Number of Chip Selects from 1 to 4
- Programmable Timing on a per Chip Select Basis
- NAND Flash Controller Supporting NAND Flash with Multiplexed Data/Address Buses
- Supports SLC and MLC NAND Flash Technology
- Supports NAND Flash Devices with 8 or 16-bit Data Paths
- Multibit Error Correcting Code (ECC) supporting NAND Flash devices with 8-bit only Data Path
- ECC Algorithm Based on Binary Shortened Bose, Chaudhuri and Hocquenghem (BCH) Codes
- Programmable Error Correcting Capability: 2, 4, 8, 12, 24 and 32 bits of Errors per Block

© 2018 Microchip Technology Inc.

- 9 Kbytes NFC SRAM
- Programmable Block Size: 512 bytes or 1024 bytes
- Programmable Number of Blocks per Page: 1, 2, 4 or 8 Blocks of Data per Page
- Programmable Spare Area Size up to 512 bytes
- Supports Spare Area ECC Protection
- Supports 8 Kbytes Page Size Using 1024 bytes/block and 4 Kbytes Page Size Using 512 bytes/block
- Multibit Error Detection Is Interrupt Driven
- Provides Hardware Acceleration for Determining Roots of Polynomials Defined over a Finite Field
- Programmable Finite Field GF(2^13) or GF(2^14)
- Finds Roots of Error-locator Polynomial
- Programmable Number of Roots
- Register Write Protection

## 36.3 Block Diagram

**Figure 36-1. Block Diagram**




## 36.4 I/O Lines Description

**Table 36-1. I/O Line Description**

| Name | Description | Type | Active Level |
|---|---|---|---|
| NCS[3:0] | Static Memory Controller Chip Select Lines | Output | Low |
| NRD | Read Signal | Output | Low |
| NWR0/NWE | Write 0/Write Enable Signal | Output | Low |
| A0/NBS0 | Address Bit 0/Byte 0 Select Signal | Output | Low |
| NWR1/NBS1 | Write 1/Byte 1 Select Signal | Output | Low |
| A[25:1] | Address Bus | Output | – |

.....continued

| Name | Description | Type | Active Level |
|---|---|---|---|
| D[15:0] | Data Bus | I/O | – |
| NWAIT | External Wait Signal | Input | Low |
| NANDRDY | NAND Flash Ready/Busy | Input | – |
| NANDWE | NAND Flash Write Enable | Output | Low |
| NANDOE | NAND Flash Output Enable | Output | Low |
| NANDALE | NAND Flash Address Latch Enable | Output | – |
| NANDCLE | NAND Flash Command Latch Enable | Output | – |

## 36.5 Multiplexed Signals

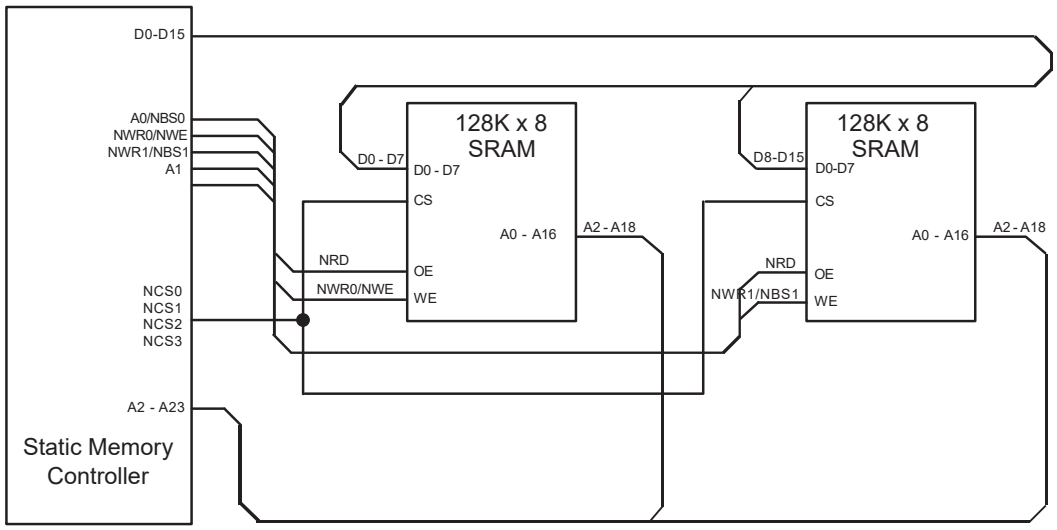
**Table 36-2. Static Memory Controller (SMC) Multiplexed Signals**

| Multiplexed Signals | | Related Function |
|---|---|---|
| NWR0 | NWE | Byte-write or Byte-select access, see "Byte Write Access" and "Byte Select Access" |
| A0 | NBS0 | 8-bit or 16-bit data bus, see "Data Bus Width" |
| A22 | NANDCLE | NAND Flash Command Latch Enable |
| A21 | NANDALE | NAND Flash Address Latch Enable |
| NWR1 | NBS1 | Byte-write or Byte-select access, see "Byte Write Access" and "Byte Select Access" |
| A1 | – | 8-/16-bit data bus, see "Data Bus Width"<br>Byte-write or Byte-select access, see "Byte Write Access" and "Byte Select Access" |

## 36.6 Application Example

### 36.6.1 Hardware Interface

**Figure 36-2. SMC Connections to Static Memory Devices**




## 36.7 Product Dependencies

### 36.7.1 I/O Lines

The pins used for interfacing the Static Memory Controller are multiplexed with the PIO lines. The programmer must first program the PIO controller to assign the Static Memory Controller pins to their peripheral function. If I/O lines of the SMC are not used by the application, they can be used for other purposes by the PIO controller.

### 36.7.2 Power Management

The SMC is clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the SMC clock.

### 36.7.3 Interrupt Sources

The SMC has an interrupt line connected to the interrupt controller. Handling the SMC interrupt requires programming the interrupt controller before configuring the SMC.

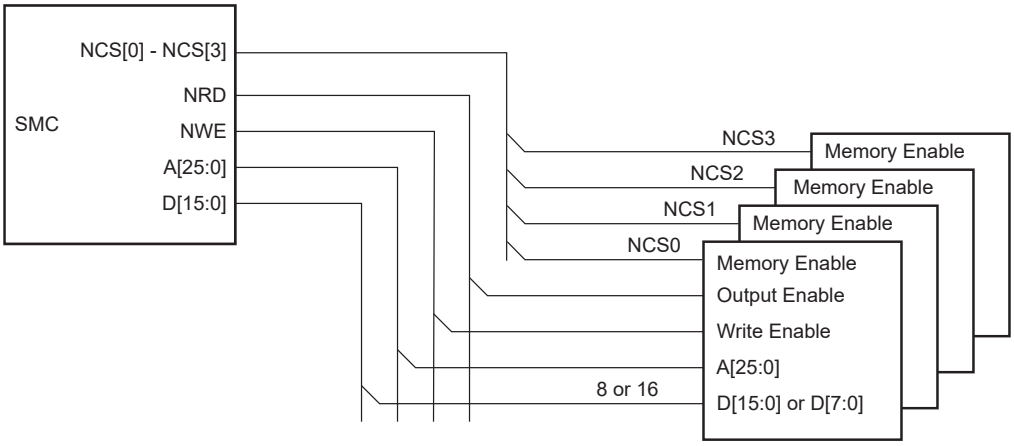
## 36.8 External Memory Mapping

The SMC provides up to 26 address lines, A[25:0]. This allows each chip select line to address up to 64 Mbytes of memory.

If the physical memory device connected on one chip select is smaller than 64 Mbytes, it wraps around and appears to be repeated within this space. The SMC correctly handles any valid access to the memory device within the page. See the figure below.

A[25:0] is only significant for 8-bit memory; A[25:1] is used for 16-bit memory.

**Figure 36-3. Memory Connections for External Devices**




## 36.9 Connection to External Devices

### 36.9.1 Data Bus Width

A data bus width of 8 or 16 bits can be selected for each chip select. This option is controlled by the bit DBW in the SMC Mode Register (HSMC_MODE) for the corresponding chip select.

"Memory Connection for an 8-bit Data Bus" shows how to connect a 512 KB x 8-bit memory on NCS2.
"Memory Connection for a 16-bit Data Bus" shows how to connect a 512 KB x 16-bit memory on NCS2.

**Figure 36-4. Memory Connection for an 8-bit Data Bus**

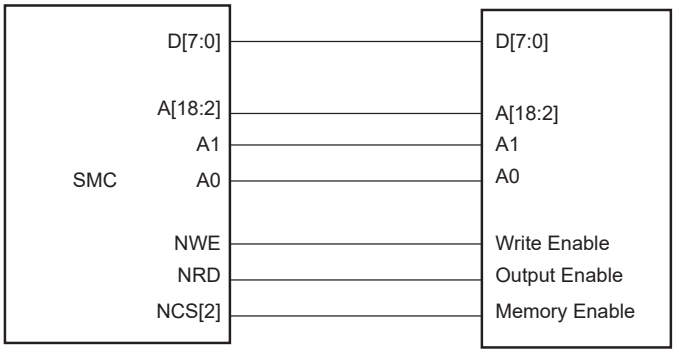
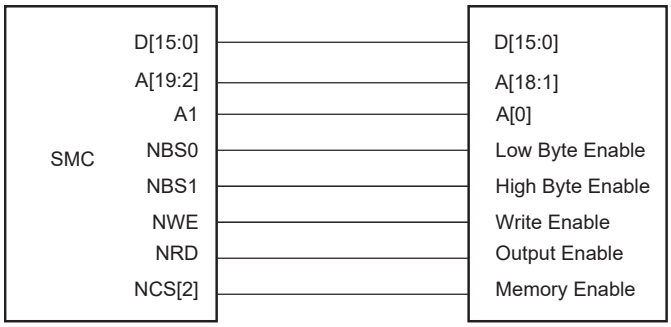



**Figure 36-5. Memory Connection for a 16-bit Data Bus**



### 36.9.2 Byte Write or Byte Select Access

Each chip select with a 16-bit data bus can operate with one of two different types of write access: Byte Write or Byte Select. This is controlled by the BAT bit of the HSMC_MODE register for the corresponding chip select.

#### 36.9.2.1 Byte Write Access

Byte Write Access is used to connect 2 x 8-bit devices as a 16-bit memory, and supports one write signal per byte of the data bus and a single read signal.

Note that the SMC does not allow boot in Byte Write Access mode.

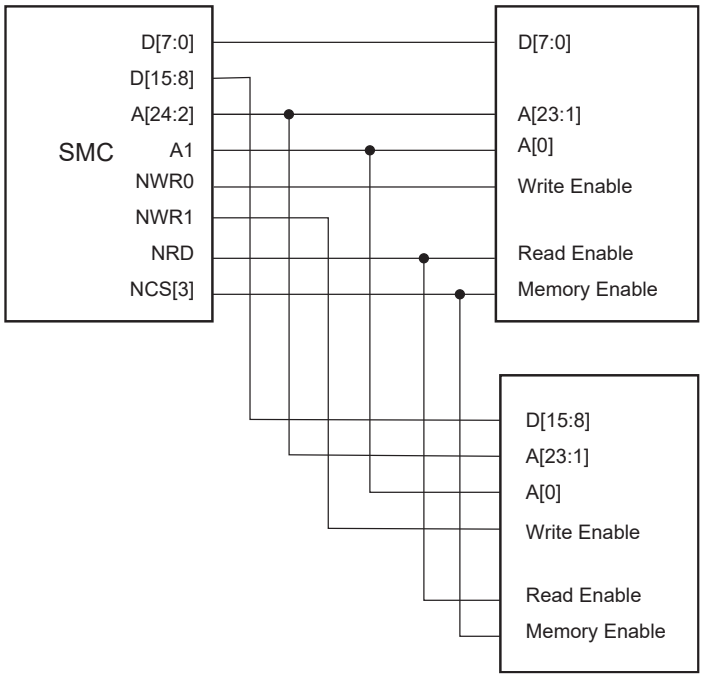
For 16-bit devices, the SMC provides NWR0 and NWR1 write signals for respectively Byte0 (lower byte) and Byte1 (upper byte) of a 16-bit bus. One single read signal (NRD) is provided.

#### 36.9.2.2 Byte Select Access

Byte Select Access is used to connect one 16-bit device. In this mode, read/write operations can be enabled/disabled at Byte level. One Byte-select line per byte of the data bus is provided. One NRD and one NWE signal control read and write.

For 16-bit devices, the SMC provides NBS0 and NBS1 selection signals for respectively Byte0 (lower byte) and Byte1 (upper byte) of a 16-bit bus.

**Figure 36-6. Connection of 2 x 8-bit Devices on a 16-bit Bus: Byte Write Option**




#### 36.9.2.3 Signal Multiplexing

Depending on the Byte Access Type (BAT), only the write signals or the byte select signals are used. To save IOs at the external bus interface, control signals at the SMC interface are multiplexed. The table below shows signal multiplexing depending on the data bus width and the Byte Access Type.

For 16-bit devices, bit A0 of address is unused. When Byte Select Option is selected, NWR1 is unused. When Byte Write option is selected, NBS0 is unused.

**Table 36-3. SMC Multiplexed Signal Translation**

| Device Type | Signal Name | | |
|---|---|---|---|
| | 16-bit Bus | | 8-bit Bus |
| | 1 x 16-bit | 2 x 8-bit | 1 x 8-bit |
| Byte Access Type (BAT) | Byte Select | Byte Write | – |
| NBS0_A0 | NBS0 | – | A0 |
| NWE_NWR0 | NWE | NWR0 | NWE |
| NBS1_NWR1 | NBS1 | NWR1 | – |
| A1 | A1 | A1 | A1 |

## 36.10 Standard Read and Write Protocols

In the following sections, the Byte Access Type is not considered. Byte select lines (NBS0 to NBS1) always have the same timing as the A address bus. NWE represents either the NWE signal in byte select access type or one of the byte write lines (NWR0 to NWR1) in byte write access type. NWR0 to NWR1 have the same timings and protocol as NWE. In the same way, NCS represents one of the NCS[0..3] chip select lines.

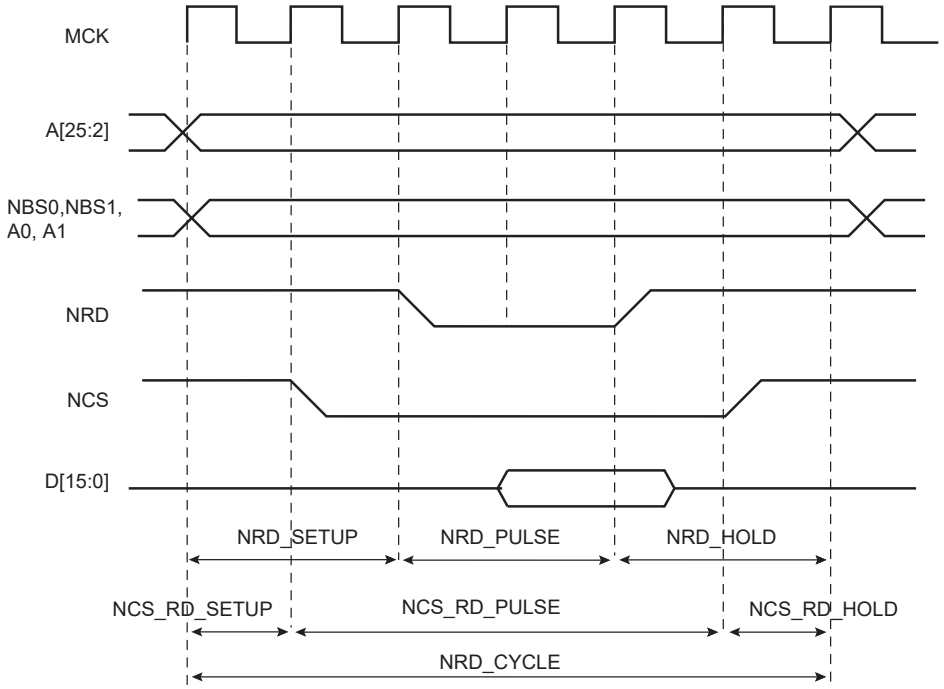
### 36.10.1 Read Waveforms

The read cycle is shown in the figure below.

The read cycle starts with the address setting on the memory address bus, i.e.,:

- {A[25:2], A1, A0} for 8-bit devices
- {A[25:2], A1} for 16-bit devices

**Figure 36-7. Standard Read Cycle**



#### 36.10.1.1 NRD Waveform

The NRD signal is characterized by a setup timing, a pulse width and a hold timing:

1. NRD_SETUP: The NRD setup time is defined as the setup of address before the NRD falling edge.
2. NRD_PULSE: The NRD pulse length is the time between NRD falling edge and NRD rising edge.
3. NRD_HOLD: The NRD hold time is defined as the hold time of address after the NRD rising edge.

#### 36.10.1.2 NCS Waveform

Similar to the NRD signal, the NCS signal can be divided into a setup time, pulse length and hold time:

- NCS_RD_SETUP: The NCS setup time is defined as the setup time of address before the NCS falling edge.
- NCS_RD_PULSE: The NCS pulse length is the time between NCS falling edge and NCS rising edge.
- NCS_RD_HOLD: The NCS hold time is defined as the hold time of address after the NCS rising edge.

#### 36.10.1.3 Read Cycle

The NRD_CYCLE time is defined as the total duration of the read cycle, that is, from the time where address is set on the address bus to the point where address may change. The total read cycle time is defined as:

- NRD_CYCLE = NRD_SETUP + NRD_PULSE + NRD_HOLD,

as well as

- NRD_CYCLE = NCS_RD_SETUP + NCS_RD_PULSE + NCS_RD_HOLD

All NRD and NCS timings are defined separately for each chip select as an integer number of Master Clock cycles. The NRD_CYCLE field is common to both the NRD and NCS signals, thus the timing period is of the same duration.

NRD_CYCLE, NRD_SETUP, and NRD_PULSE implicitly define the NRD_HOLD value as:

- NRD_HOLD = NRD_CYCLE - NRD SETUP - NRD PULSE

NRD_CYCLE, NCS_RD_SETUP, and NCS_RD_PULSE implicitly define the NCS_RD_HOLD value as:

- NCS_RD_HOLD = NRD_CYCLE - NCS_RD_SETUP - NCS_RD_PULSE

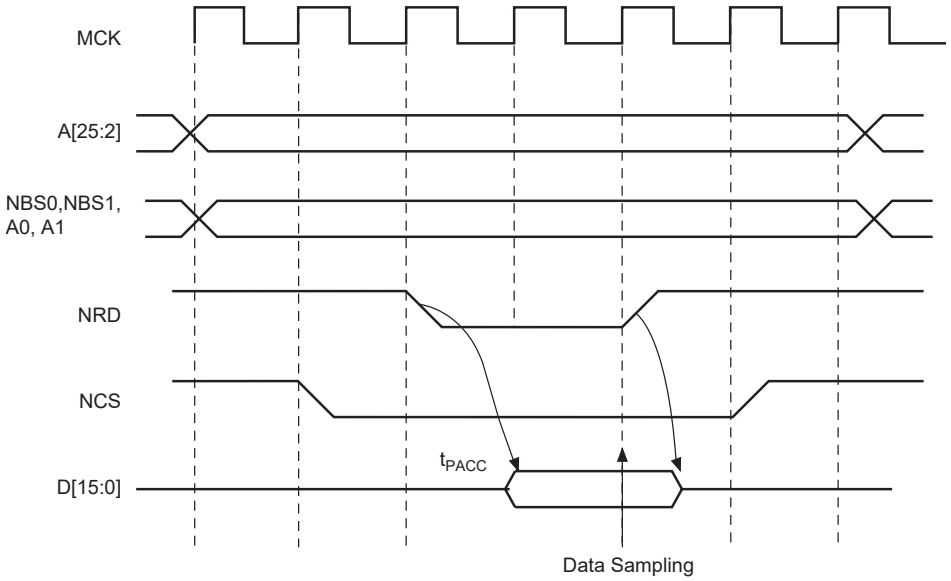
### 36.10.2 Read Mode

As NCS and NRD waveforms are defined independently of one another, the SMC needs to know when the read data is available on the data bus. The SMC does not compare NCS and NRD timings to know which signal rises first. The READ_MODE parameter in the HSMC_MODE register of the corresponding chip select indicates which signal of NRD and NCS controls the read operation.

#### 36.10.2.1 Read is Controlled by NRD (READ_MODE = 1)

The figure below shows the waveforms of a read operation of a typical asynchronous RAM. The read data is available $t_{PACC}$ after the falling edge of NRD, and turns to ‘Z’ after the rising edge of NRD. In this case, the READ_MODE must be set to 1 (read is controlled by NRD), to indicate that data is available with the rising edge of NRD. The SMC samples the read data internally on the rising edge of the Master Clock that generates the rising edge of NRD, whatever the programmed waveform of NCS.

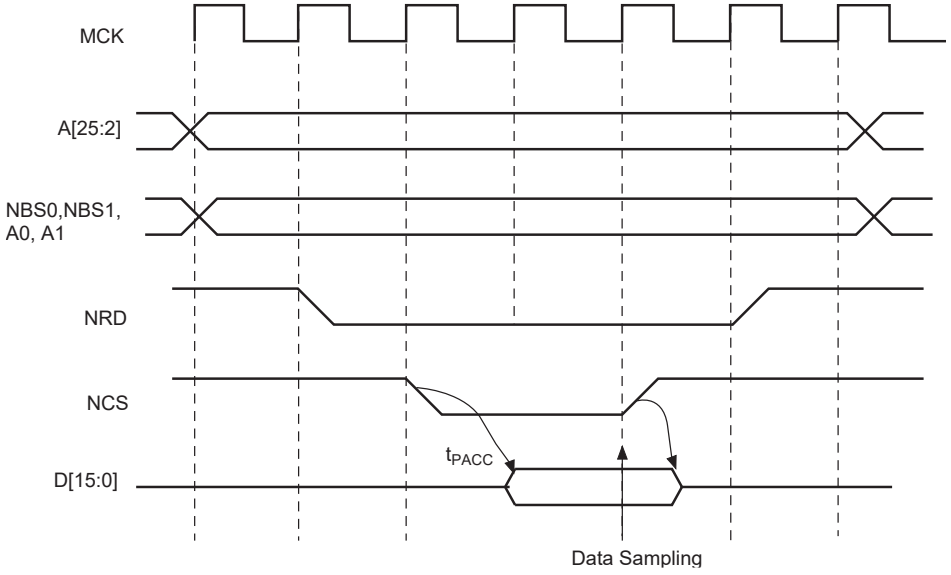
**Figure 36-8. READ_MODE = 1: Data is Sampled by SMC before the Rising Edge of NRD**




#### 36.10.2.2 Read is Controlled by NCS (READ_MODE = 0)

The figure below shows the typical read cycle. The read data is valid $t_{PACC}$ after the falling edge of the NCS signal and remains valid until the rising edge of NCS. Data must be sampled when NCS is raised. In that case, the READ_MODE must be configured to 0 (read is controlled by NCS): the SMC internally samples the data on the rising edge of the Master Clock that generates the rising edge of NCS, whatever the programmed waveform of NRD.

**Figure 36-9. READ_MODE = 0: Data is Sampled by SMC before the Rising Edge of NCS**




### 36.10.3 Write Waveforms

The write protocol is similar to the read protocol. It is depicted in the figure below. The write cycle starts with the address setting on the memory address bus.

#### 36.10.3.1 NWE Waveforms

The NWE signal is characterized by a setup timing, a pulse width and a hold timing:

- NWE_SETUP: The NWE setup time is defined as the setup of address and data before the NWE falling edge.
- NWE_PULSE: The NWE pulse length is the time between NWE falling edge and NWE rising edge.
- NWE_HOLD: The NWE hold time is defined as the hold time of address and data after the NWE rising edge.

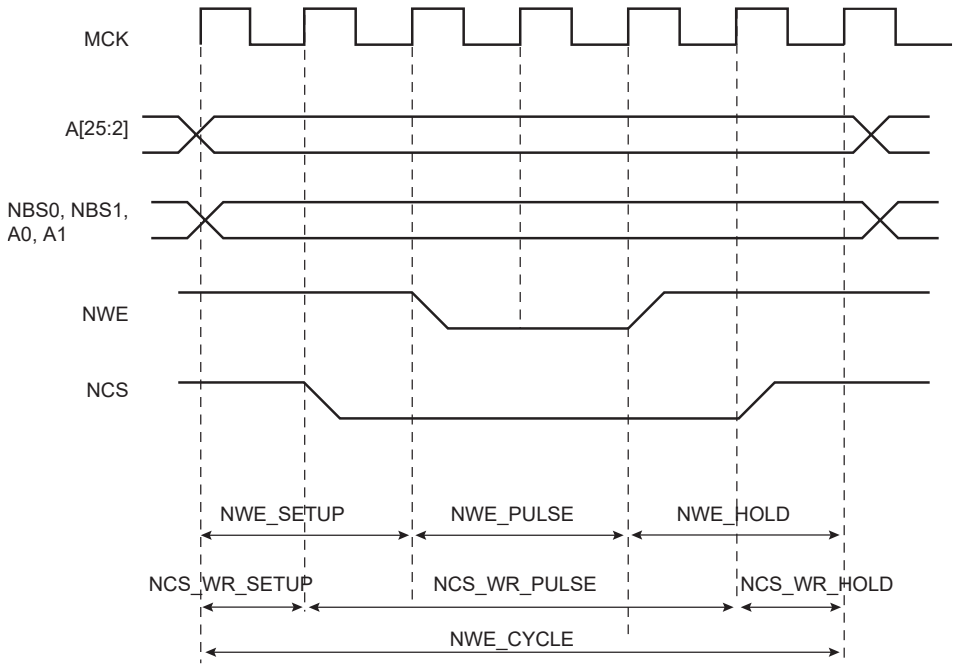
The NWE waveforms apply to all byte-write lines in Byte Write Access mode: NWR0 to NWR3.

#### 36.10.3.2 NCS Waveforms

The NCS signal waveforms in write operations are not the same as those applied in read operations, but are separately defined:

- NCS_WR_SETUP: The NCS setup time is defined as the setup time of address before the NCS falling edge.
- NCS_WR_PULSE: The NCS pulse length is the time between NCS falling edge and NCS rising edge.
- NCS_WR_HOLD: The NCS hold time is defined as the hold time of address after the NCS rising edge.

**Figure 36-10. Write Cycle**




#### 36.10.3.3 Write Cycle

The write cycle time is defined as the total duration of the write cycle, that is, from the time where address is set on the address bus to the point where address may change. The total write cycle time is equal to:

- NWE_CYCLE = NWE_SETUP + NWE_PULSE + NWE_HOLD,

as well as

- NWE_CYCLE = NCS_WR_SETUP + NCS_WR_PULSE + NCS_WR_HOLD

All NWE and NCS (write) timings are defined separately for each chip select as an integer number of Master Clock cycles. The NWE_CYCLE field is common to both the NWE and NCS signals, thus the timing period is of the same duration.

NWE_CYCLE, NWE_SETUP, and NWE_PULSE implicitly define the NWE_HOLD value as:

- NWE_HOLD = NWE_CYCLE - NWE_SETUP - NWE_PULSE

NWE_CYCLE, NCS_WR_SETUP, and NCS_WR_PULSE implicitly define the NCS_WR_HOLD value as:

- NCS_WR_HOLD = NWE_CYCLE - NCS_WR_SETUP - NCS_WR_PULSE

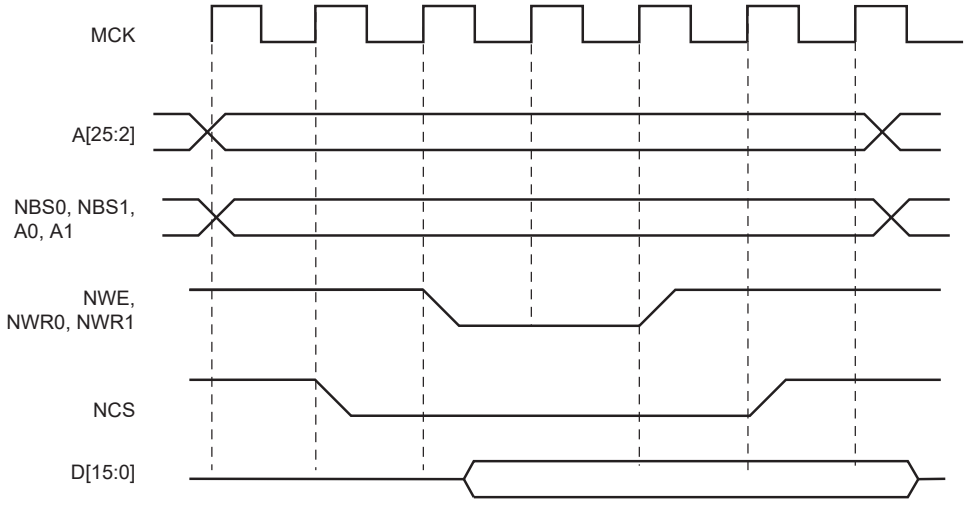
### 36.10.4 Write Mode

The WRITE_MODE parameter in the HSMC_MODE register of the corresponding chip select indicates which signal controls the write operation.

#### 36.10.4.1 Write is Controlled by NWE (WRITE_MODE = 1)

The figure below shows the waveforms of a write operation with WRITE_MODE set to 1. The data is put on the bus during the pulse and hold steps of the NWE signal. The internal data buffers are switched to Output mode after the NWE_SETUP time, and until the end of the write cycle, regardless of the programmed waveform on NCS.

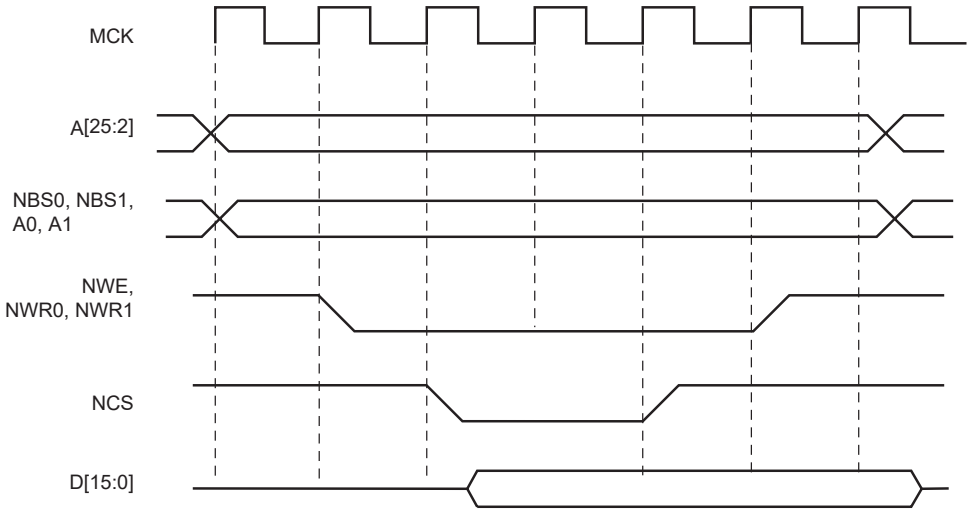
**Figure 36-11. WRITE_MODE = 1. The write operation is controlled by NWE**




#### 36.10.4.2 Write is Controlled by NCS (WRITE_MODE = 0)

The figure below shows the waveforms of a write operation with WRITE_MODE configured to 0. The data is put on the bus during the pulse and hold steps of the NCS signal. The internal data buffers are switched to Output mode after the NCS_WR_SETUP time, and until the end of the write cycle, regardless of the programmed waveform on NWE.

**Figure 36-12. WRITE_MODE = 0. The write operation is controlled by NCS**




### 36.10.5 Coding Timing Parameters

All timing parameters are defined for one chip select and are grouped together in one register according to their type:

- The HSMC_SETUP register groups the definition of all setup parameters: NRD_SETUP, NCS_RD_SETUP, NWE_SETUP, NCS_WR_SETUP
- The HSMC_PULSE register groups the definition of all pulse parameters: NRD_PULSE, NCS_RD_PULSE, NWE_PULSE, NCS_WR_PULSE
- The HSMC_CYCLE register groups the definition of all cycle parameters: NRD_CYCLE, NWE_CYCLE

The table below shows how the timing parameters are coded and their permitted range.

**Table 36-4. Coding and Range of Timing Parameters**

| Coded Value | Number of Bits | Effective Value | Permitted Range | |
|---|---|---|---|---|
| | | | Coded Value | Effective Value |
| setup [5:0] | 6 | 128 x setup[5] + setup[4:0] | 0 ≤ setup ≤ 31 | 0..31 |
| | | | 32 ≤ setup ≤ 63 | 128..(128 + 31) |
| pulse [6:0] | 7 | 256 x pulse[6] + pulse[5:0] | 0 ≤ pulse ≤ 63 | 0..63 |
| | | | 64 ≤ pulse ≤ 127 | 256..(256 + 63) |
| cycle[8:0] | 9 | 256 x cycle[8:7] + cycle[6:0] | 0 ≤ cycle ≤ 127 | 0..127 |
| | | | 128 ≤ cycle ≤ 255 | 256..(256 + 127) |
| | | | 256 ≤ cycle ≤ 383 | 512..(512 + 127) |
| | | | 384 ≤ cycle ≤ 511 | 768..(768 + 127) |

### 36.10.6 Reset Values of Timing Parameters

The table below gives the default value of timing parameters at reset.

**Table 36-5. Reset Values of Timing Parameters**

| Register | Reset Value | Description |
|---|---|---|
| HSMC_SETUP | 0x01010101 | All setup timings are set to 1 |
| HSMC_PULSE | 0x01010101 | All pulse timings are set to 1 |
| HSMC_CYCLE | 0x00030003 | The read and write operations last three Master Clock cycles and provide one hold cycle. |
| WRITE_MODE | 1 | Write is controlled with NWE |
| READ_MODE | 1 | Read is controlled with NRD |

### 36.10.7 Usage Restriction

The SMC does not check the validity of the user-programmed parameters. If the sum of SETUP and PULSE parameters is larger than the corresponding CYCLE parameter, this leads to an unpredictable behavior of the SMC.

#### 36.10.7.1 For Read Operations

Null but positive setup and hold of address and NRD and/or NCS cannot be guaranteed at the memory interface because of the propagation delay of these signals through external logic and pads. When positive setup and hold values must be verified, then it is strictly recommended to program non-null values so as to cover possible skews between address, NCS and NRD signals.

#### 36.10.7.2 For Write Operations

If a null hold value is programmed on NWE, the SMC can guarantee a positive hold of address, byte select lines, and NCS signal after the rising edge of NWE. This is true for WRITE_MODE = 1 only. See “Early Read Wait State”.

#### 36.10.7.3 For Read and Write Operations

A null value for pulse parameters is forbidden and may lead to an unpredictable behavior.

In read and write cycles, the setup and hold time parameters are defined in reference to the address bus. For external devices that require setup and hold time between NCS and NRD signals (read), or between NCS and NWE signals (write), these setup and hold times must be converted into setup and hold times in reference to the address bus.

## 36.11 Scrambling/Unscrambling Function

The external data bus D[15:0] can be scrambled in order to prevent intellectual property data located in off-chip memories from being easily recovered by analyzing data at the package pin level of either the microcontroller or the memory device.

The scrambling and unscrambling are performed on-the-fly without additional wait states.

The scrambling method depends on two user-configurable key registers, HSMC_KEY1 and HSMC_KEY2. These key registers are only accessible in Write mode.

The key must be securely stored in a reliable nonvolatile memory in order to recover data from the off-chip memory. Any data scrambled with a given key cannot be recovered if the key is lost.

The scrambling/unscrambling function is enabled or disabled by configuring specific bits in the HSMC_OCMS and the HSMC_TIMINGSx registers. The bit configuration values to enable memory scrambling are summarized in the table below.

**Table 36-6. Scrambling Function Bit Encoding**

| Memories | Bit Values | | |
|---|---|---|---|
| | HSMC_OCMS.SMSE | HSMC_OCMS.SRSE | HSMC_TIMINGSx.OCMS |
| Off-chip Memories | 1 | 0 | 1 |
| NAND Flash with NFC | 0 | 1 | 0 |

When the NAND Flash memory content is scrambled, the on-chip NFC SRAM page buffer associated for the transfer is also scrambled.

## 36.12 Automatic Wait States

Under certain circumstances, the SMC automatically inserts idle cycles between accesses to avoid bus contention or operation conflict.

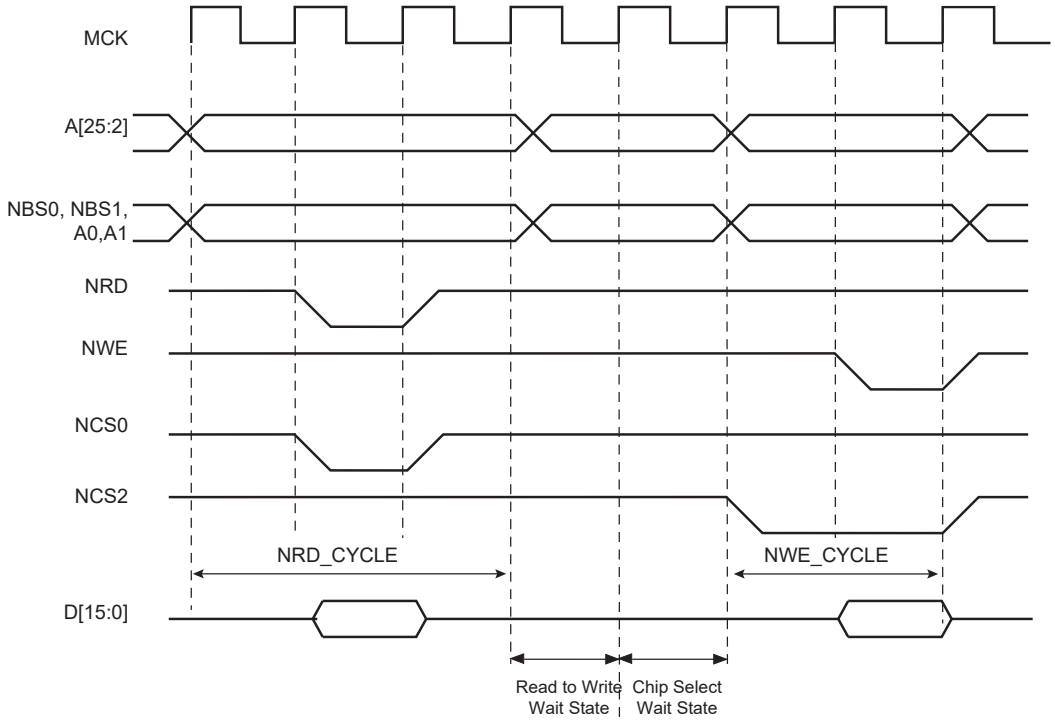
### 36.12.1 Chip Select Wait States

The SMC always inserts an idle cycle between two transfers on separate chip selects. This idle cycle ensures that there is no bus contention between the deactivation of one device and the activation of the next one.

During chip select wait state, all control lines are turned inactive: NBS0 to NBS1, NWR0 to NWR1, NCS[0..3], and NRD lines. They are all set to 1.

The figure below illustrates a chip select wait state between access on Chip Select 0 and Chip Select 2.

**Figure 36-13. Chip Select Wait State between a Read Access on NCS0 and a Write Access on NCS2**



### 36.12.2 Early Read Wait State

In some cases, the SMC inserts a wait state cycle between a write access and a read access to allow time for the write cycle to end before the subsequent read cycle begins. This wait state is not generated in addition to a chip select wait state. The early read cycle thus only occurs between a write and read access to the same memory device (same chip select).

An early read wait state is automatically inserted if at least one of the following conditions is valid:

- if the write controlling signal has no hold time and the read controlling signal has no setup time (see figure "Early Read Wait State: Write with No Hold Followed by Read with No Setup").
- in NCS Write Controlled mode (WRITE_MODE = 0), if there is no hold timing on the NCS signal and the NCS_RD_SETUP parameter is configured to 0, regardless of the Read mode (see figure "Early Read Wait State: NCS Controlled Write with No Hold Followed by a Read with No NCS Setup"). The write operation must end with an NCS rising edge. Without an Early Read Wait State, the write operation could not complete properly.
- in NWE Controlled mode (WRITE_MODE = 1) and if there is no hold timing (NWE_HOLD = 0), the feedback of the write control signal is used to control address, data, chip select and byte select lines. If the external write control signal is not inactivated as expected due to load capacitances, an Early Read Wait State is inserted and address, data and control signals are maintained one more cycle. See figure "Early Read Wait State: NWE-controlled Write with No Hold Followed by a Read with One Setup Cycle".

**Figure 36-14. Early Read Wait State: Write with No Hold Followed by Read with No Setup**

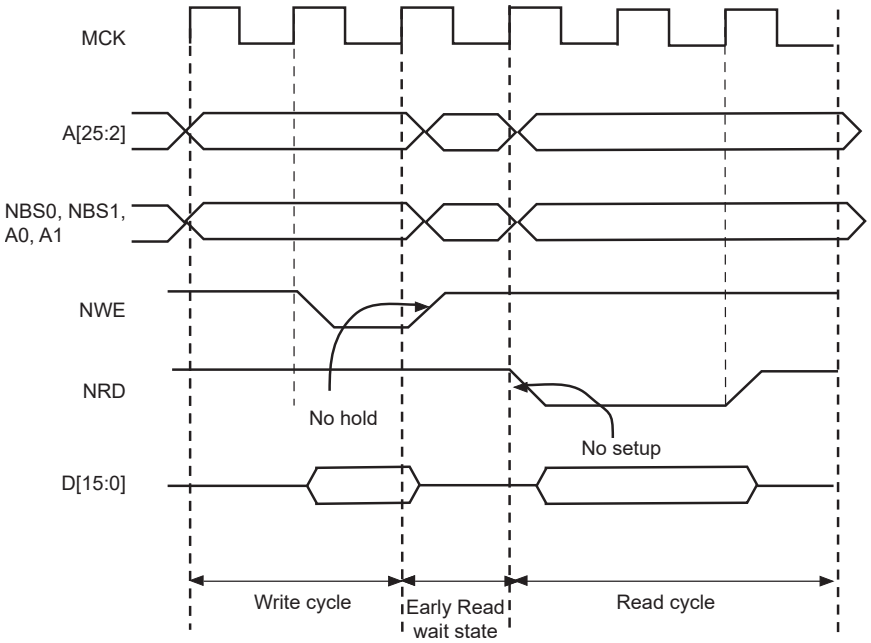


**Figure 36-15. Early Read Wait State: NCS Controlled Write with No Hold Followed by a Read with No NCS Setup**

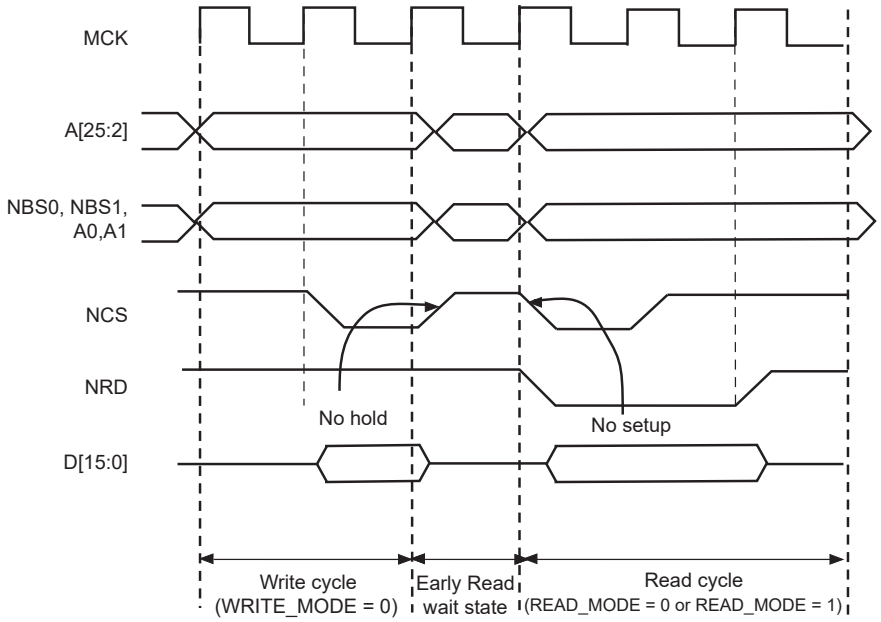
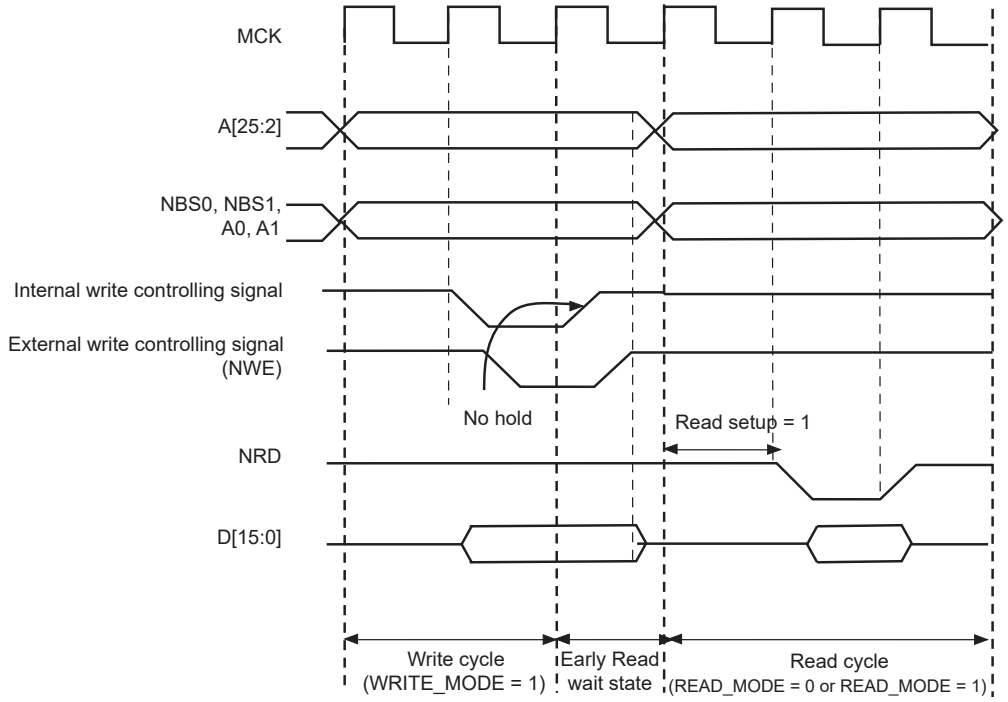



**Figure 36-16. Early Read Wait State: NWE-controlled Write with No Hold Followed by a Read with One Setup Cycle**




### 36.12.3 Reload User Configuration Wait State

The user may change any of the configuration parameters by writing the SMC user interface.

When detecting that a new user configuration has been written in the user interface, the SMC inserts a wait state before starting the next access. The so called “Reload User Configuration Wait State” is used by the SMC to load the new set of parameters to apply to next accesses.

The Reload Configuration Wait State is not applied in addition to the Chip Select Wait State. If accesses before and after reprogramming the user interface are made to different devices (Chip Selects), then one single Chip Select Wait State is applied.

On the other hand, if accesses before and after writing the user interface are made to the same device, a Reload Configuration Wait State is inserted, even if the change does not concern the current Chip Select.

#### 36.12.3.1 User Procedure

To insert a Reload Configuration Wait State, the SMC detects a write access to any HSMC_MODE register of the user interface. If only the timing registers are modified (HSMC_SETUP, HSMC_PULSE, HSMC_CYCLE registers) in the user interface, the user must validate the modification by writing the HSMC_MODE register, even if no change was made on the mode parameters.

#### 36.12.3.2 Slow Clock Mode Transition

A Reload Configuration Wait State is also inserted when the Slow Clock Mode is entered or exited, after the end of the current transfer (see “Slow Clock Mode”).

### 36.12.4 Read to Write Wait State

Due to an internal mechanism, a wait cycle is always inserted between consecutive read and write SMC accesses.

This wait cycle is referred to as a read to write wait state in this document.

This wait cycle is applied in addition to chip select and reload user configuration wait states when they are to be inserted. See figure “Chip Select Wait State between a Read Access on NCS0 and a Write Access on NCS2”.

## 36.13 Data Float Wait States

Some memory devices are slow to release the external bus. For such devices, it is necessary to add wait states (data float wait states) after a read access:

- before starting a read access to a different external memory
- before starting a write access to the same device or to a different external one.

The Data Float Output Time ($t_{DF}$) for each external memory device is programmed in the TDF_CYCLES field of the HSMC_MODE register for the corresponding chip select. The value of TDF_CYCLES indicates the number of data float wait cycles (between 0 and 15) before the external device releases the bus, and represents the time allowed for the data output to go to high impedance after the memory is disabled.

Data float wait states do not delay internal memory accesses. Hence, a single access to an external memory with long $t_{DF}$ will not slow down the execution of a program from internal memory.

The data float wait states management depends on the READ_MODE and the TDF_MODE bits of the HSMC_MODE register for the corresponding chip select.

### 36.13.1 READ_MODE

Setting READ_MODE to 1 indicates to the SMC that the NRD signal is responsible for turning off the tri-state buffers of the external memory device. The Data Float Period then begins after the rising edge of the NRD signal and lasts TDF_CYCLES MCK cycles.

When the read operation is controlled by the NCS signal (READ_MODE = 0), the TDF_CYCLES field in HSMC_MODEx gives the number of MCK cycles during which the data bus remains busy after the rising edge of NCS.

The figure "TDF Period in NRD Controlled Read Access (TDF = 2)" illustrates the Data Float Period in NRD-controlled mode (READ_MODE = 1), assuming a data float period of two cycles (TDF_CYCLES = 2). The figure "TDF Period in NCS Controlled Read Operation (TDF = 3)" shows the read operation when controlled by NCS (READ_MODE = 0) and the TDF_CYCLES parameter equals 3.

**Figure 36-17. TDF Period in NRD Controlled Read Access (TDF = 2)**

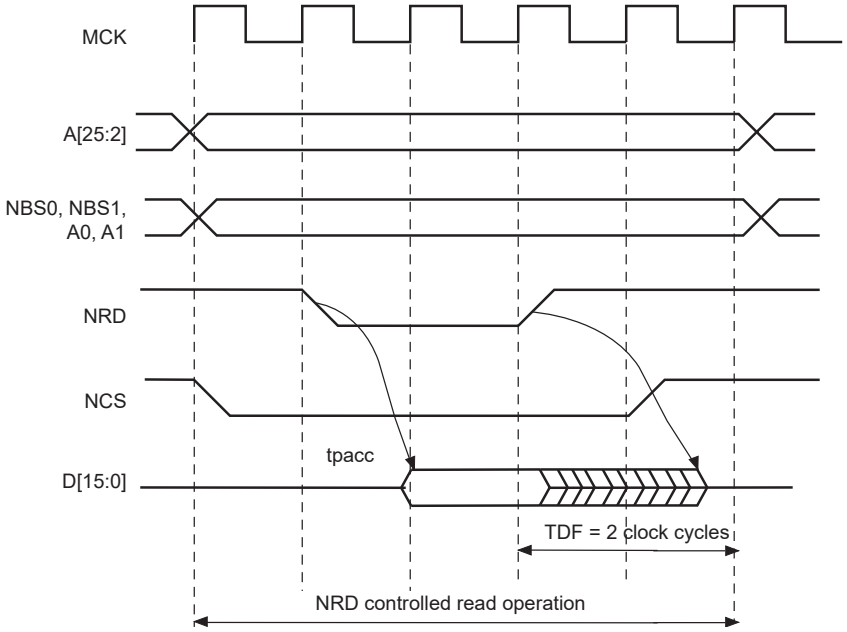
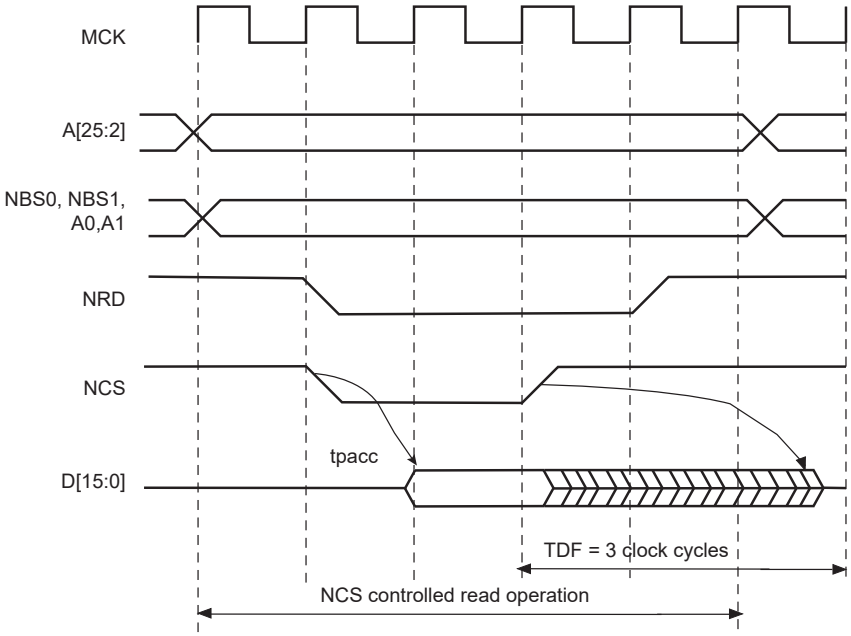



**Figure 36-18. TDF Period in NCS Controlled Read Operation (TDF = 3)**




### 36.13.2 TDF Optimization Enabled (TDF_MODE = 1)

When the TDF_MODE of the HSMC_MODE register is set to 1 (TDF optimization is enabled), the SMC takes advantage of the setup period of the next access to optimize the number of wait states cycle to insert.

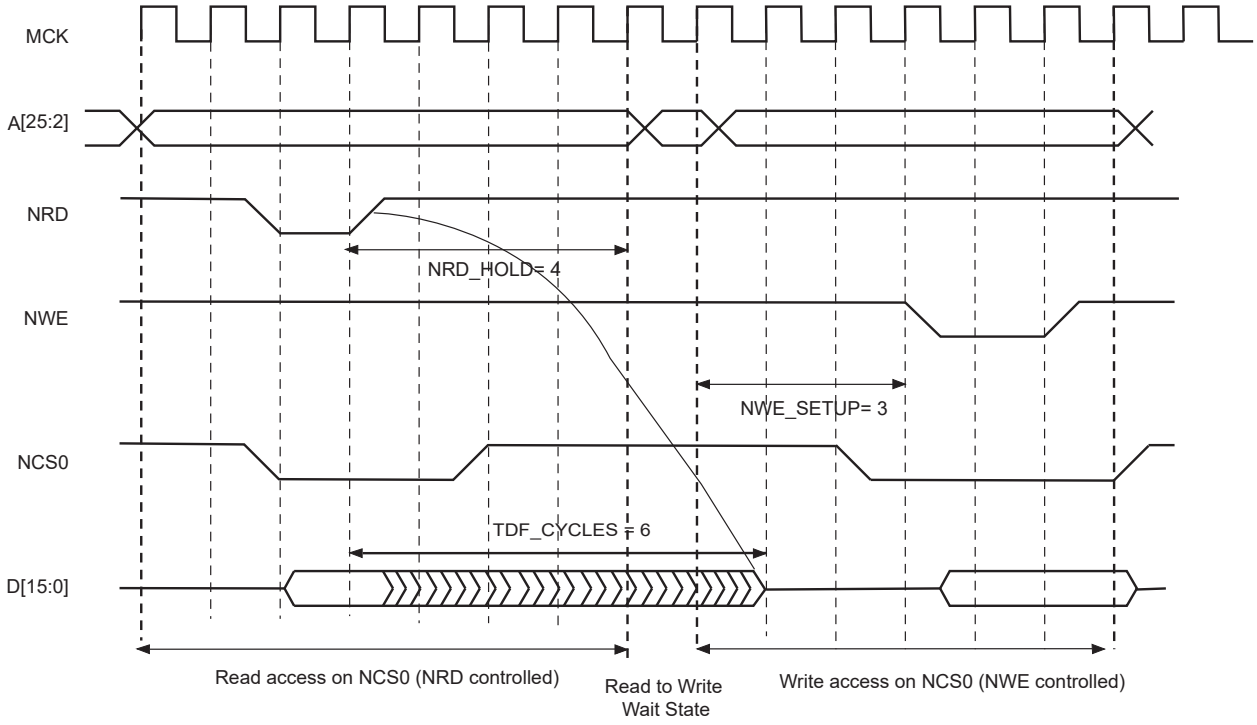
The figure below shows a read access controlled by NRD, followed by a write access controlled by NWE, on Chip Select 0. Chip Select 0 has been programmed with:

NRD_HOLD = 4; READ_MODE = 1 (NRD controlled)

NWE_SETUP = 3; WRITE_MODE = 1 (NWE controlled)

TDF_CYCLES = 6; TDF_MODE = 1 (optimization enabled).

**Figure 36-19. TDF Optimization: No TDF wait states are inserted if the TDF period is over when the next access begins**




### 36.13.3 TDF Optimization Disabled (TDF_MODE = 0)

When optimization is disabled, TDF wait states are inserted at the end of the read transfer, so that the data float period ends when the second access begins. If the hold period of the read1 controlling signal overlaps the data float period, no additional TDF wait states will be inserted.

The figures below illustrate the cases:

- read access followed by a read access on another chip select,
- read access followed by a write access on another chip select,
- read access followed by a write access on the same chip select,

with no TDF optimization.

**Figure 36-20. TDF Optimization Disabled (TDF Mode = 0). TDF wait states between 2 read accesses on different chip selects**

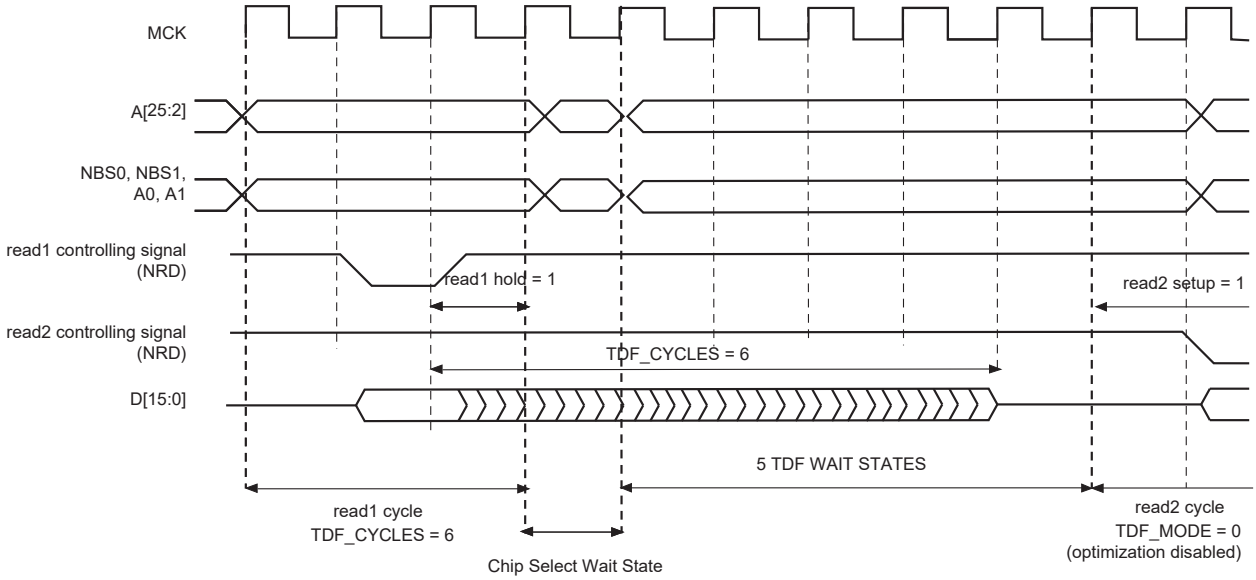



**Figure 36-21. TDF Mode = 0: TDF wait states between a read and a write access on different chip selects**

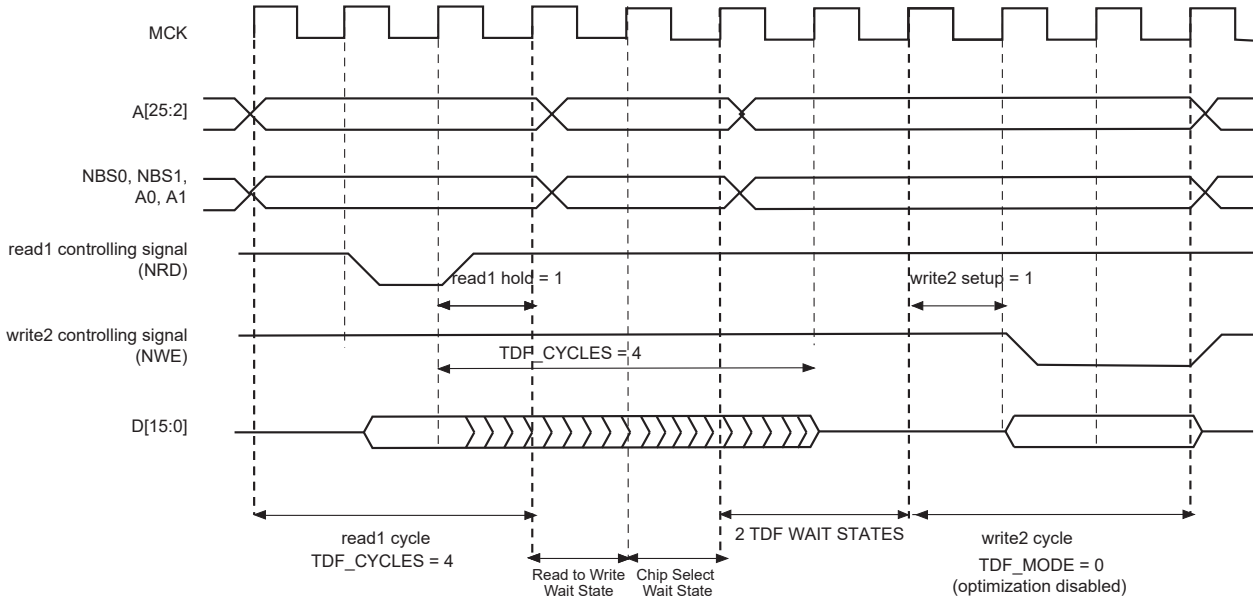
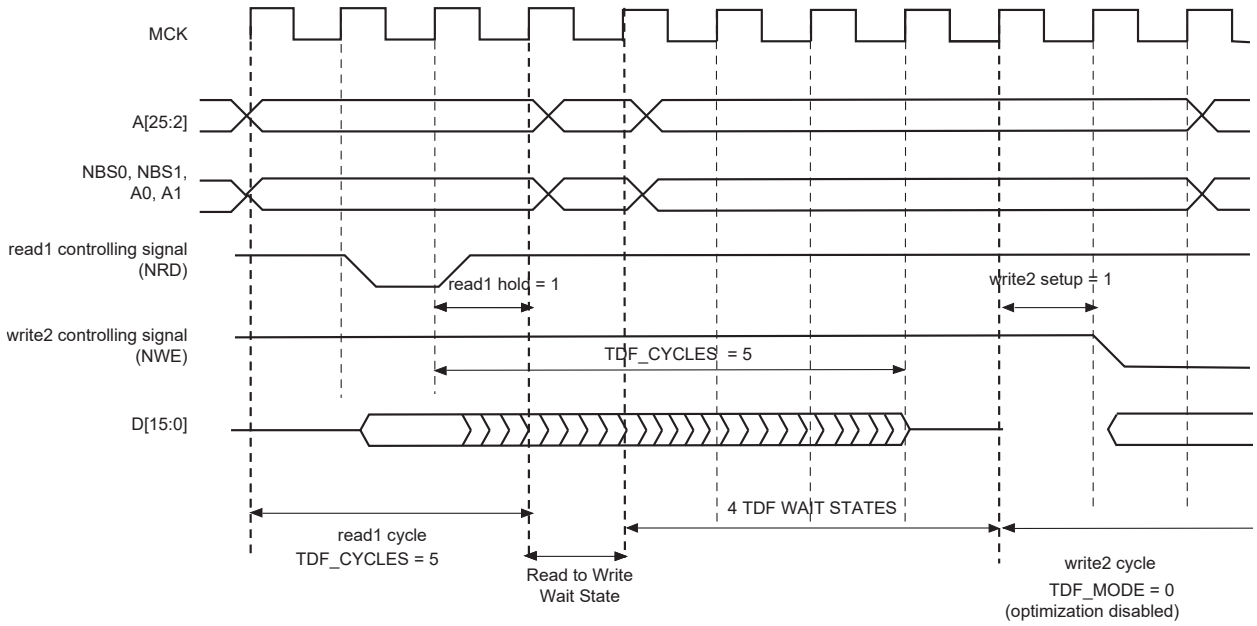


**Figure 36-22. TDF Mode = 0: TDF wait states between read and write accesses on the same chip select**




## 36.14 External Wait

Any access can be extended by an external device using the NWAIT input signal of the SMC. The EXNW_MODE field of the HSMC_MODE register on the corresponding chip select must be set to either ‘10’ (Frozen mode) or ‘11’ (Ready mode). When the EXNW_MODE is set to ‘00’ (disabled), the NWAIT signal is simply ignored on the corresponding chip select. The NWAIT signal delays the read or write operation in regards to the read or write controlling signal, depending on the Read and Write modes of the corresponding chip select.

### 36.14.1 Restriction

When one of the EXNW_MODE is enabled, it is mandatory to program at least one hold cycle for the read/write controlling signal. For that reason, the NWAIT signal cannot be used in Slow Clock Mode (see “Slow Clock Mode”).

The NWAIT signal is assumed to be a response of the external device to the read/write request of the SMC. NWAIT is then examined by the SMC in the pulse state of the read or write controlling signal. The assertion of the NWAIT signal outside the expected period has no impact on the SMC behavior.

### 36.14.2 Frozen Mode

When the external device asserts the NWAIT signal (active low), and after an internal synchronization of this signal, the SMC state is frozen, i.e., SMC internal counters are frozen, and all control signals remain unchanged. When the resynchronized NWAIT signal is deasserted, the SMC completes the access, resuming the access from the point where it was stopped. See the figure below. This mode must be selected when the external device uses the NWAIT signal to delay the access and to freeze the SMC.

The assertion of the NWAIT signal outside the expected period is ignored as illustrated in the figure “Read Access with NWAIT Assertion in Frozen Mode (EXNW_MODE = 10)”.

**Figure 36-23. Write Access with NWAIT Assertion in Frozen Mode (EXNW_MODE = 10)**

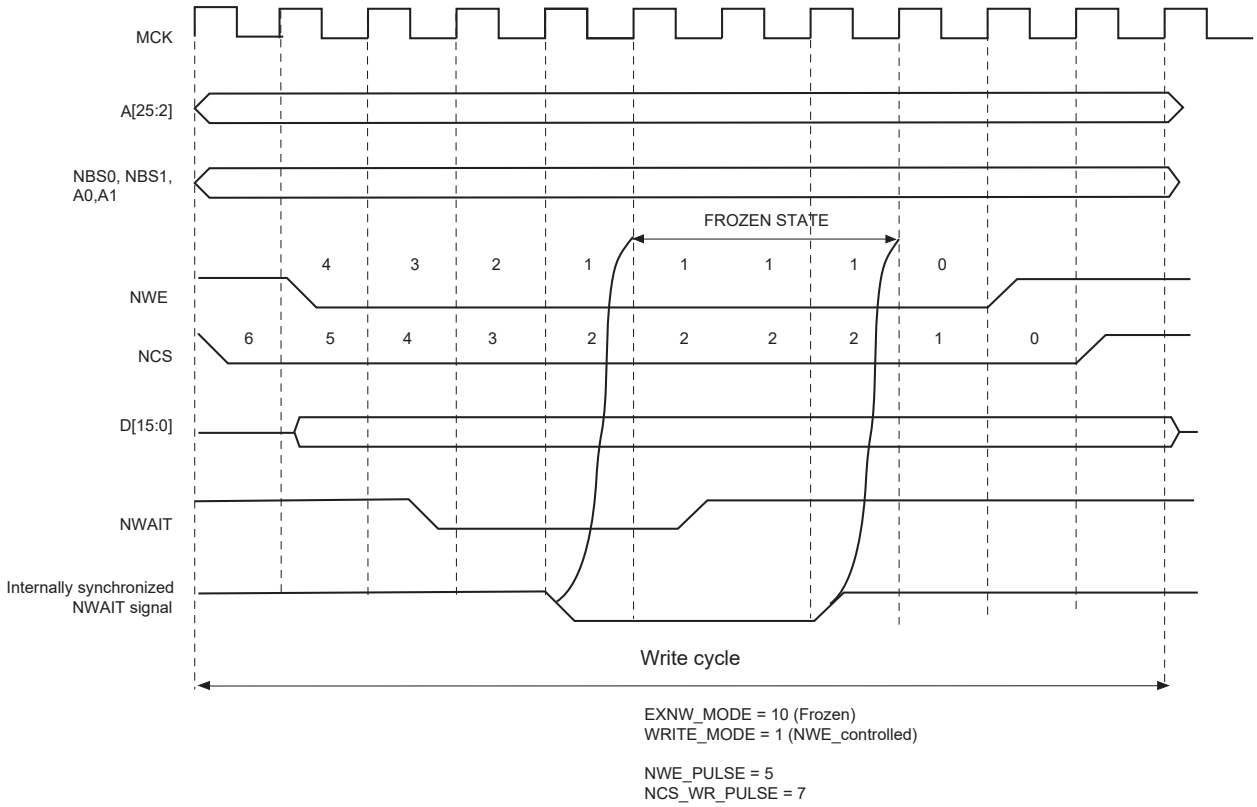
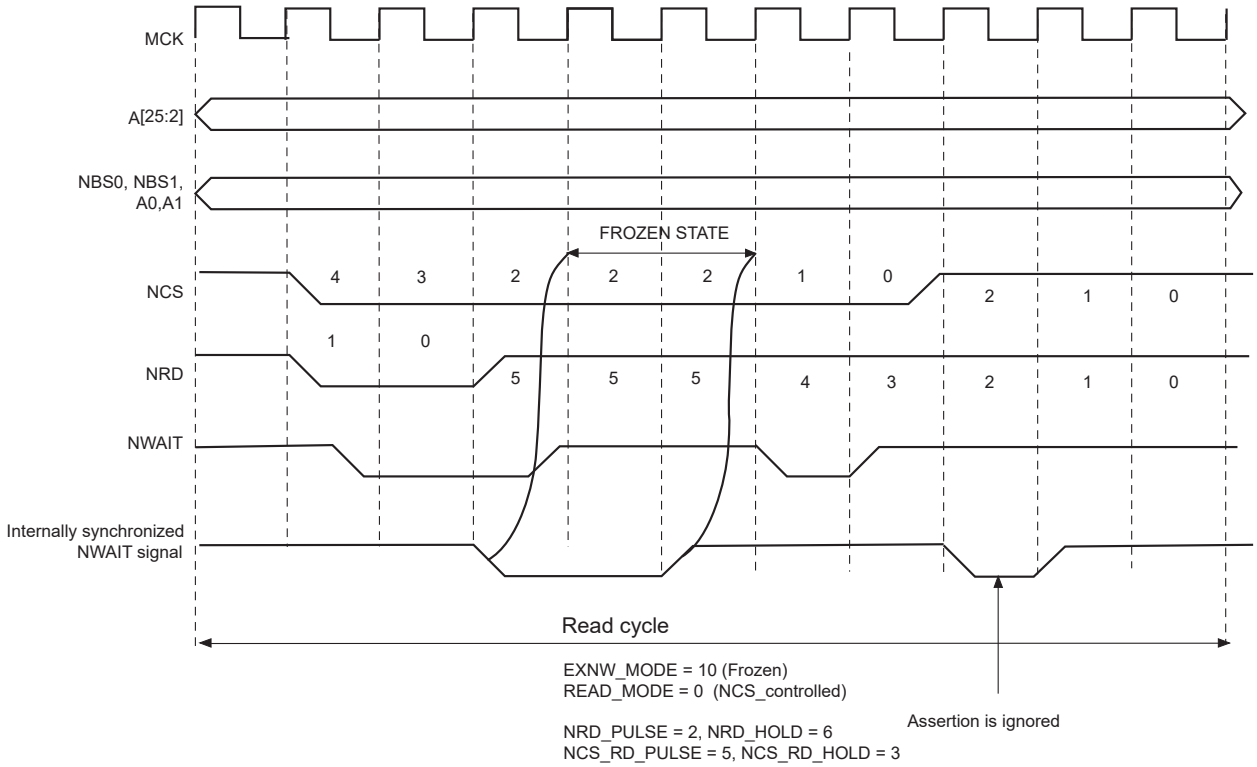



**Figure 36-24. Read Access with NWAIT Assertion in Frozen Mode (EXNW_MODE = 10)**



### 36.14.3 Ready Mode

In Ready mode (EXNW_MODE = 11), the SMC behaves differently. Normally, the SMC begins the access by down counting the setup and pulse counters of the read/write controlling signal. In the last cycle of the pulse phase, the resynchronized NWAIT signal is examined.

If asserted, the SMC suspends the access as shown in the figures below. After deassertion, the access is completed: the hold step of the access is performed.

This mode must be selected when the external device uses deassertion of the NWAIT signal to indicate its ability to complete the read or write operation.

If the NWAIT signal is deasserted before the end of the pulse, or asserted after the end of the pulse of the controlling read/write signal, it has no impact on the access length as shown in the figure “NWAIT Assertion in Read Access: Ready Mode (EXNW_MODE = 11)”.

**Figure 36-25. NWAIT Assertion in Write Access: Ready Mode (EXNW_MODE = 11)**

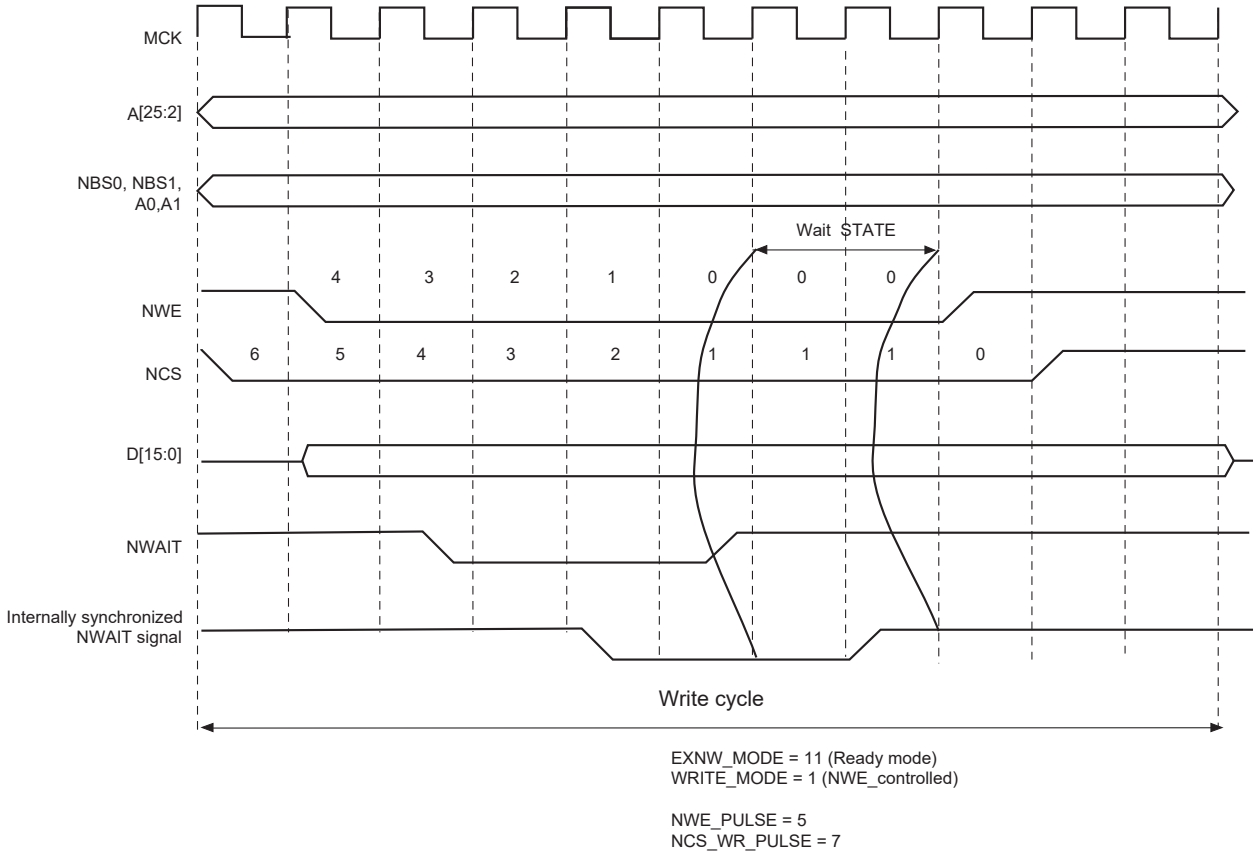
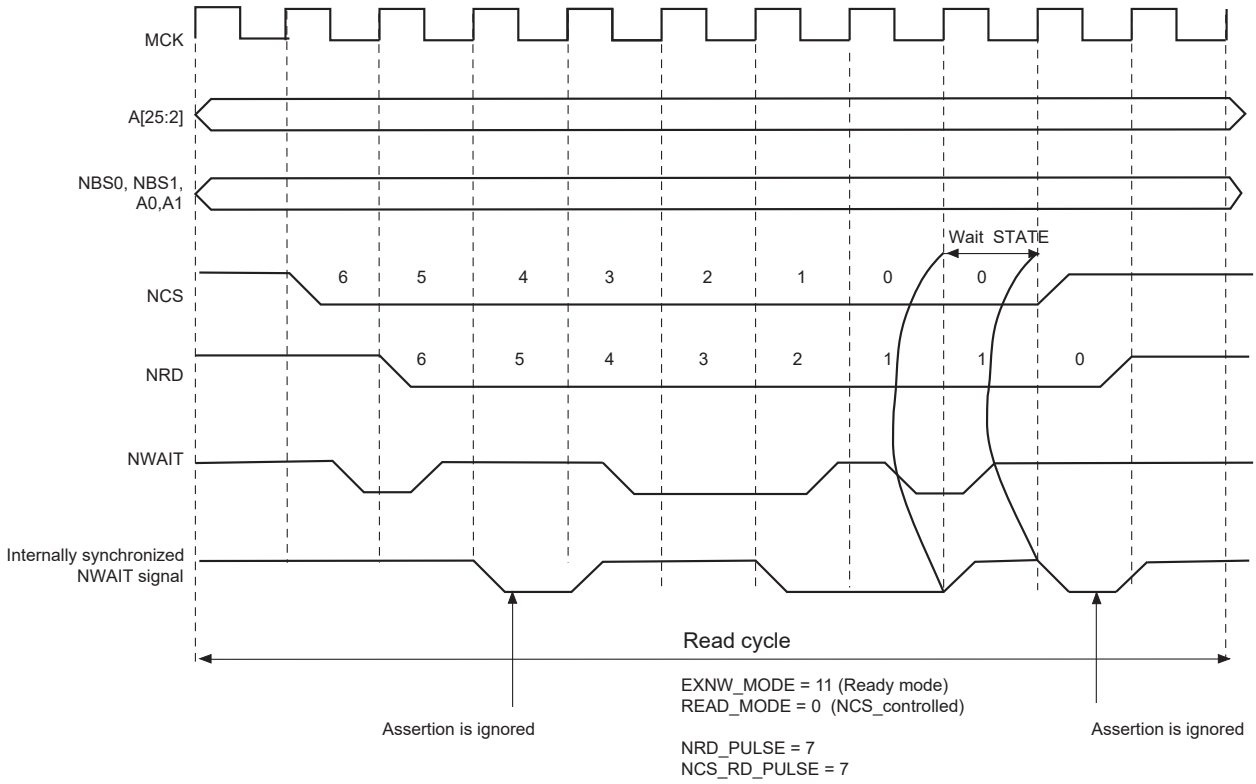


**Figure 36-26. NWAIT Assertion in Read Access: Ready Mode (EXNW_MODE = 11)**



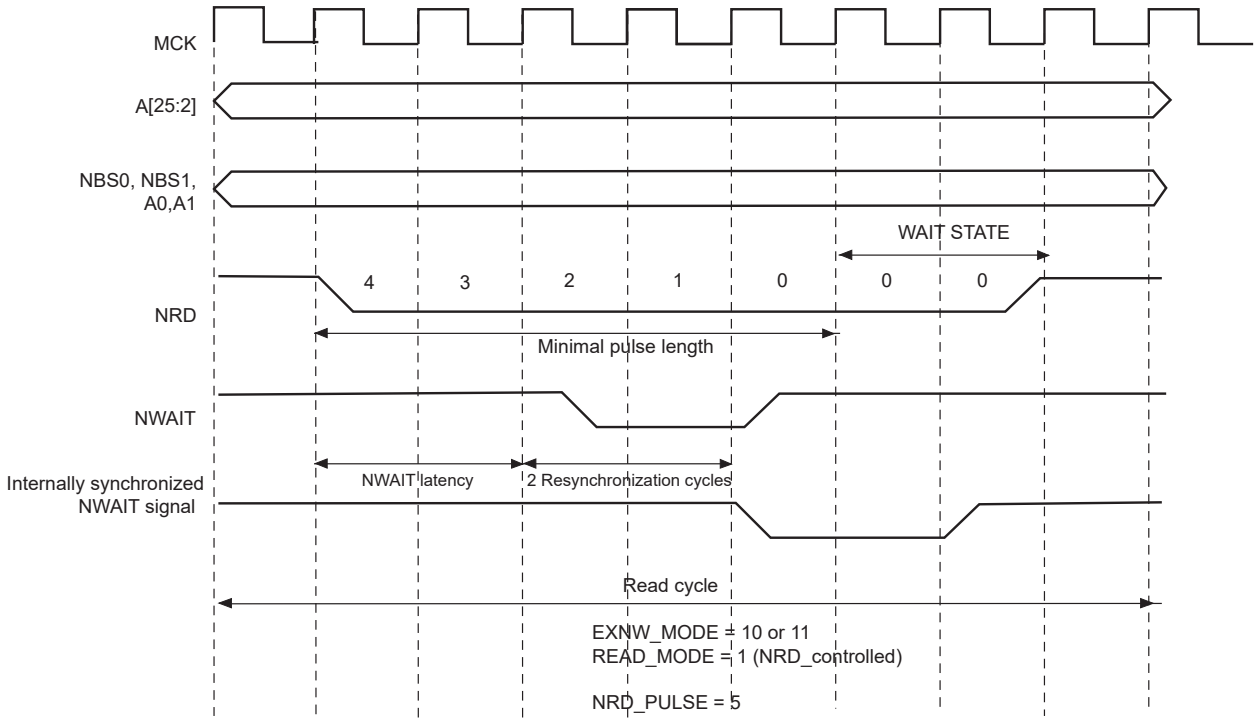

### 36.14.4 NWAIT Latency and Read/Write Timings

There may be a latency between the assertion of the read/write controlling signal and the assertion of the NWAIT signal by the device. The programmed pulse length of the read/write controlling signal must be at least equal to this latency plus the 2 cycles of resynchronization + 1 cycle. Otherwise, the SMC may enter the hold state of the access without detecting the NWAIT signal assertion. This is true in Frozen mode as well as in Ready mode. This is illustrated in the figure below.

When EXNW_MODE is enabled (ready or frozen), the user must program a pulse length of the read and write controlling signal of at least:

minimal pulse length = NWAIT latency + 2 resynchronization cycles + 1 cycle

**Figure 36-27. NWAIT Latency**




## 36.15 Slow Clock Mode

The SMC is able to automatically apply a set of “Slow Clock mode” read/write waveforms when an internal signal driven by the Power Management Controller is asserted because MCK has been turned to a very slow clock rate (typically 32 kHz clock rate). In this mode, the user-programmed waveforms are ignored and the Slow Clock mode waveforms are applied. This mode is provided so as to avoid reprogramming the User Interface with appropriate waveforms at very slow clock rate. When activated, the Slow mode is active on all chip selects.

### 36.15.1 Slow Clock Mode Waveforms

The figure below illustrates the read and write operations in Slow Clock mode. They are valid on all chip selects. The table below indicates the value of read and write parameters in Slow Clock mode.

**Figure 36-28. Write/Read Cycles in Slow Clock Mode**

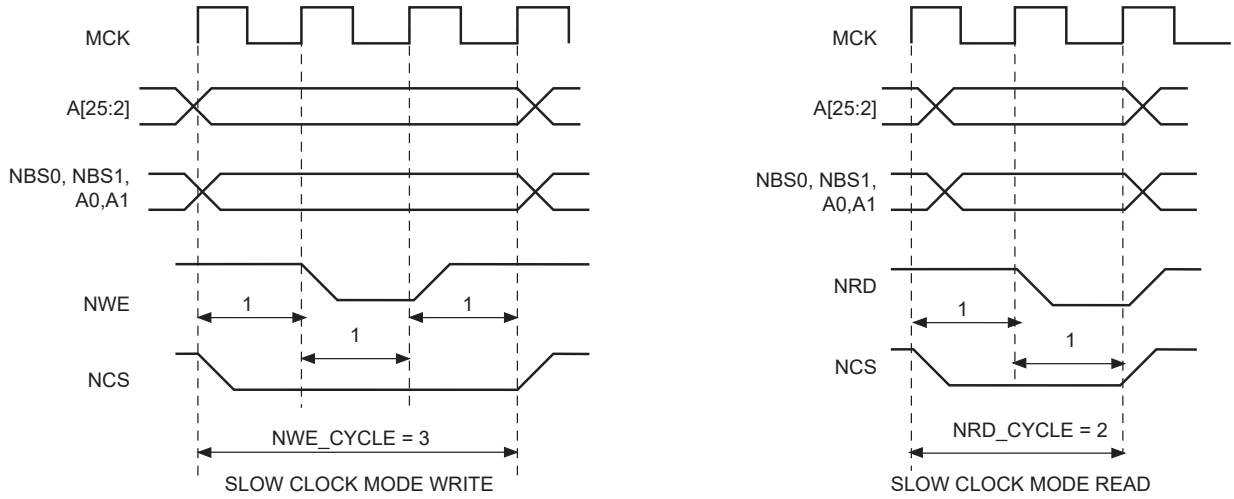



**Table 36-7. Read and Write Timing Parameters in Slow Clock Mode**

| Read Parameters | Duration (cycles) | Write Parameters | Duration (cycles) |
|---|---|---|---|
| NRD_SETUP | 1 | NWE_SETUP | 1 |
| NRD_PULSE | 1 | NWE_PULSE | 1 |
| NCS_RD_SETUP | 0 | NCS_WR_SETUP | 0 |
| NCS_RD_PULSE | 2 | NCS_WR_PULSE | 3 |
| NRD_CYCLE | 2 | NWE_CYCLE | 3 |

### 36.15.2 Switching from (to) Slow Clock Mode to (from) Normal Mode

When switching from Slow Clock mode to Normal mode, the current Slow Clock mode transfer is completed at high clock rate, with the set of Slow Clock mode parameters. See the figure "Clock Rate Transition occurs while the SMC is performing a Write Operation". The external device may not be fast enough to support such timings.

The figure "Recommended Procedure to Switch from Slow Clock Mode to Normal Mode or from Normal Mode to Slow Clock Mode" illustrates the recommended procedure to properly switch from one mode to the other.

**Figure 36-29. Clock Rate Transition occurs while the SMC is performing a Write Operation**

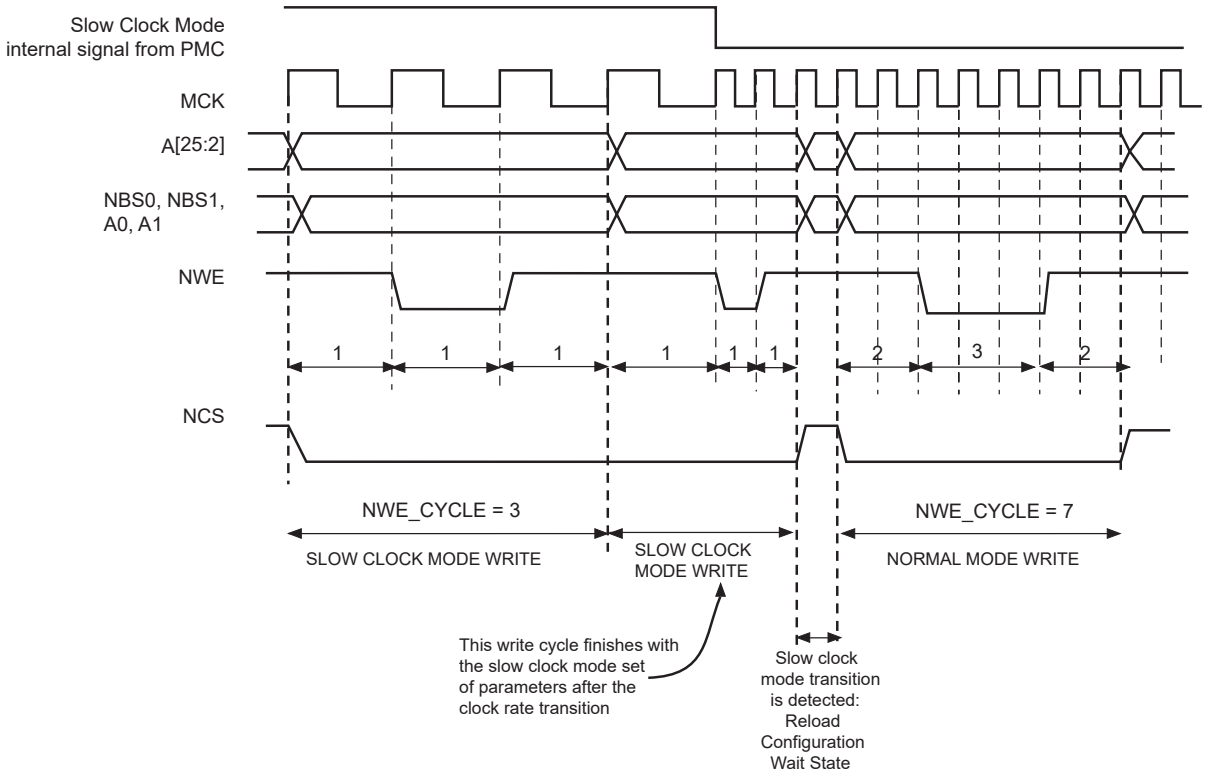
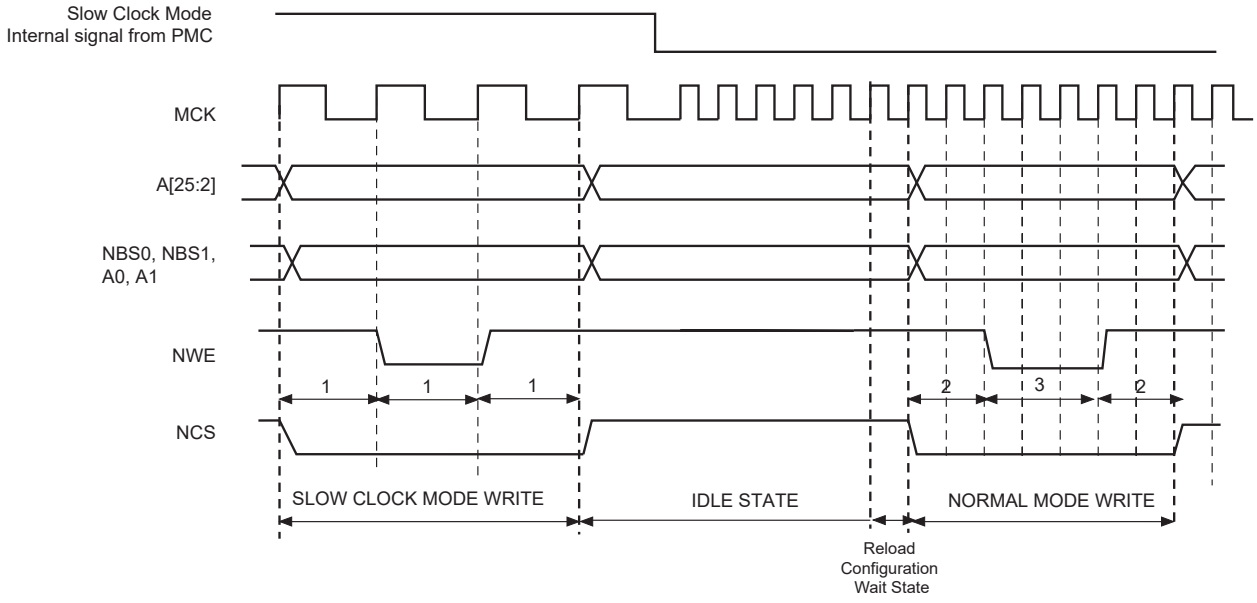



**Figure 36-30. Recommended Procedure to Switch from Slow Clock Mode to Normal Mode or from Normal Mode to Slow Clock Mode**




## 36.16 Register Write Protection

To prevent any single software error that may corrupt SMC behavior, selected registers can be write-protected by setting the WPEN bit in the Write Protection Mode Register (HSMC_WPMR).

If a write access in a write-protected register is detected, then the WPVS flag in the Write Protection Status Register (HSMC_WPSR) is set and the field WPVSRC indicates in which register the write access has been attempted.

The WPVS flag is automatically reset after reading the HSMC_WPSR.

The following registers can be write-protected:

- Setup Register
- Pulse Register
- Cycle Register
- Timings Register
- Mode Register

## 36.17 NFC Operations

### 36.17.1 NFC Overview

The NFC handles all the command, address and data sequences of the NAND low level protocol. An SRAM is used as an internal read/write buffer when data is transferred from or to the NAND.

### 36.17.2 NFC Control Registers

NAND Flash Read and NAND Flash Program operations can be performed through the NFC Command Registers. In order to minimize CPU intervention and latency, commands are posted in a command buffer. This buffer provides zero wait state latency.

The NFC handles an automatic transfer between the external NAND Flash and the chip via the NFC SRAM. The transfer is done by programming NFC Command Registers.

The NFC Command Registers are very efficient to use. When writing to these registers:

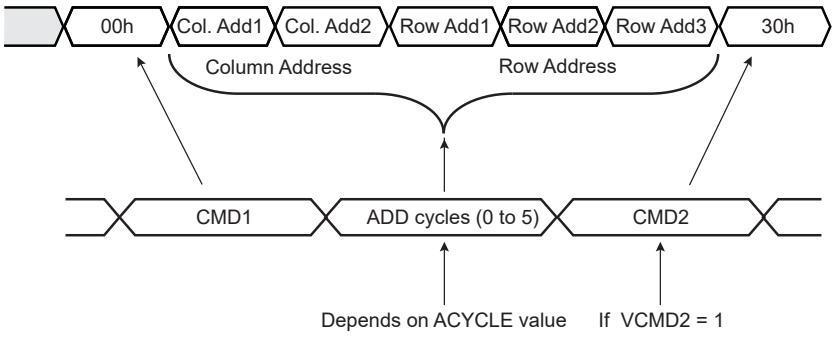
- the address of the register (NFCADDR_CMD) is the command used
- the data of the register (NFCDATA_ADDT) is the address to be sent to the NAND Flash

In one single access the command is sent and immediately executed by the NFC. Two commands can even be programmed within a single access (CMD1, CMD2) depending on the VCMD2 value.

The NFC can send up to five address cycles.

The figure below shows a typical NAND Flash Page Read Command of a NAND Flash Memory and correspondence with NFC Address Command Register.

**Figure 36-31. NFC/NAND Flash Access Example**




For more details, refer to 36.17.2.2 NFCADDR_CMD.

Reading the NFC Command Register (to any address) will give the status of the NFC. This is especially useful to know if the NFC is busy, for example.

#### 36.17.2.1 Building NFC Address Command Example

The base address is made of HOST_ADDR address.

Page read operation example:

```
// Build the Address Command (NFCADDR_CMD)
AddressCommand = (HOST_ADDR |
  NFCWR=0    | // NFC Read Data from NAND Flash
  DATAEN=1   | // NFC Data phase Enable.
  CSID=1     | // Chip Select ID = 1
  ACYCLE= 5  | // Number of address cycle.
  VCMD2=1    | // CMD2 is sent after Address Cycles
  CMD2=0x30  | // CMD2 = 30h
  CMD1=0x0) // CMD1 = Read Command = 00h
 // Set the Address for Cycle 0
 HSMC_ADDR = Col. Add1
 // Write command with the Address Command built above
*AddressCommand = (Col. Add2 | // ADDR_CYCLE1
  Row Add1   | // ADDR_CYCLE2
  Row Add2   | // ADDR_CYCLE3
  Row Add3 )   // ADDR_CYCLE4
```

#### 36.17.2.2 NFC Address Command

**Name:** NFCADDR_CMD
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | NFCWR | DATAEN | CSID[2:2] |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CSID[1:0] | | ACYCLE[2:0] | | | VCMD2 | CMD2[7:6] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMD2[5:0] | | | | | | CMD1[7:6] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMD1[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | | | | | | | | |

**Bit 26 – NFCWR** NFC Write Enable

| Value | Description |
|---|---|
| 0 | NFC reads data from the NAND Flash. |
| 1 | NFC writes data into the NAND Flash. |

**Bit 25 – DATAEN** NFC Data Phase Enable
When set to true, the NFC will automatically read or write data after the command.

**Bits 24:22 – CSID[2:0]** Chip Select Identifier
Chip select used.

**Bits 21:19 – ACYCLE[2:0]** Number of Address Required for the Current Command
When ACYCLE field is different from zero, ACYCLE Address cycles are performed after Command Cycle 1. The maximum number of cycles is 5.

**Bit 18 – VCMD2** Valid Cycle 2 Command
When set to true, the CMD2 field is issued after the address cycle.

**Bits 17:10 – CMD2[7:0]** Command Register Value for Cycle 2
When a write access occurs with the VCMD2 field set, the NFC sends this command after CMD1.

**Bits 9:2 – CMD1[7:0]** Command Register Value for Cycle 1
When a write access occurs, the NFC sends this command.

#### 36.17.2.3 NFC Data Address

**Name:** NFCDATA_ADDT
**Property:** Write-only

If five address cycles are used, the first address cycle is ADDR_CYCLE0. Refer to HSMC_ADDR register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR_CYCLE4[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR_CYCLE3[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR_CYCLE2[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR_CYCLE1[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | | | | | | | | |

**Bits 31:24 – ADDR_CYCLE4[7:0]** NAND Flash Array Address Cycle 4
When less than five address cycles are used, ADDR_CYCLE4 is the fourth byte written to the NAND Flash.
When five address cycles are used, ADDR_CYCLE4 is the fifth byte written to NAND Flash.

**Bits 23:16 – ADDR_CYCLE3[7:0]** NAND Flash Array Address Cycle 3
When less than five address cycles are used, ADDR_CYCLE3 is the third byte written to the NAND Flash.
When five address cycles are used, ADDR_CYCLE3 is the fourth byte written to NAND Flash.

**Bits 15:8 – ADDR_CYCLE2[7:0]** NAND Flash Array Address Cycle 2
When less than five address cycles are used, ADDR_CYCLE2 is the second byte written to the NAND Flash.
When five address cycles are used, ADDR_CYCLE2 is the third byte written to NAND Flash.

**Bits 7:0 – ADDR_CYCLE1[7:0]** NAND Flash Array Address Cycle 1
When less than five address cycles are used, ADDR_CYCLE1 is the first byte written to the NAND Flash.
When five address cycles are used, ADDR_CYCLE1 is the second byte written to NAND Flash.

#### 36.17.2.4 NFC Data Status

**Name:** NFCDATA_STATUS
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | NFCBUSY | NFCWR | DATAEN | CSID[2:2] |
| Access | | | | | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CSID[1:0] | | ACYCLE[2:0] | | | VCMD2 | CMD2[7:6] | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMD2[5:0] | | | | | | CMD1[7:6] | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMD1[5:0] | | | | | | | |
| Access | R | R | R | R | R | R | | |
| Reset | | | | | | | | |

**Bit 27 – NFCBUSY** NFC Busy Status Flag
If set to true, it indicates that the NFC is busy.

**Bit 26 – NFCWR** NFC Write Enable

| Value | Description |
|---|---|
| 0 | NFC is in Read mode. |
| 1 | NFC is in Write mode. |

**Bit 25 – DATAEN** NFC Data Phase Enable
When set to true, the NFC data phase is enabled.

**Bits 24:22 – CSID[2:0]** Chip Select Identifier
Chip select used.

**Bits 21:19 – ACYCLE[2:0]** Number of Address Required for the Current Command
When ACYCLE is different from zero, ACYCLE Address cycles are performed after Command Cycle 1.

**Bit 18 – VCMD2** Valid Cycle 2 Command
When set to true, the CMD2 field is issued after the address cycle.

**Bits 17:10 – CMD2[7:0]** Command Register Value for Cycle 2
When VCMD2 bit is set to true, the Physical Memory Interface drives the IO bus with CMD2 field during the Command Latch cycle 2.

**Bits 9:2 – CMD1[7:0]** Command Register Value for Cycle 1
When a Read or Write Access occurs, the Physical Memory Interface drives the IO bus with CMD1 field during the Command Latch cycle 1.

### 36.17.3 NFC Initialization

Prior to any Command and Data Transfer, the SMC User Interface must be configured to meet the device timing requirements.

- Write Enable Configuration

Use NWE_SETUP, NWE_PULSE and NWE_CYCLE to define the write enable waveform according to the external device datasheet.

Use HSMC_TIMINGS.TADL to configure the timing between the last address latch cycle and the first rising edge of WEN for data input.

**Figure 36-32. Write Enable Timing Configuration**

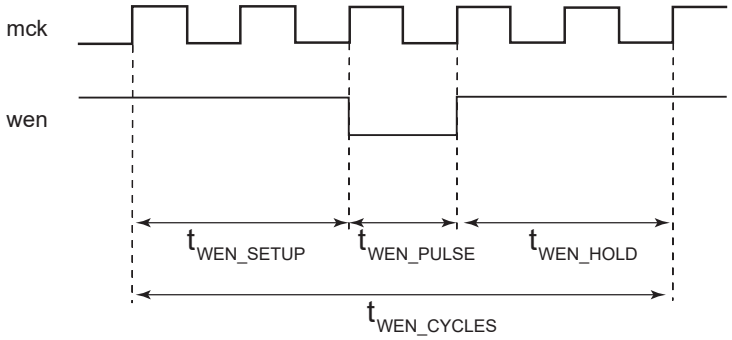
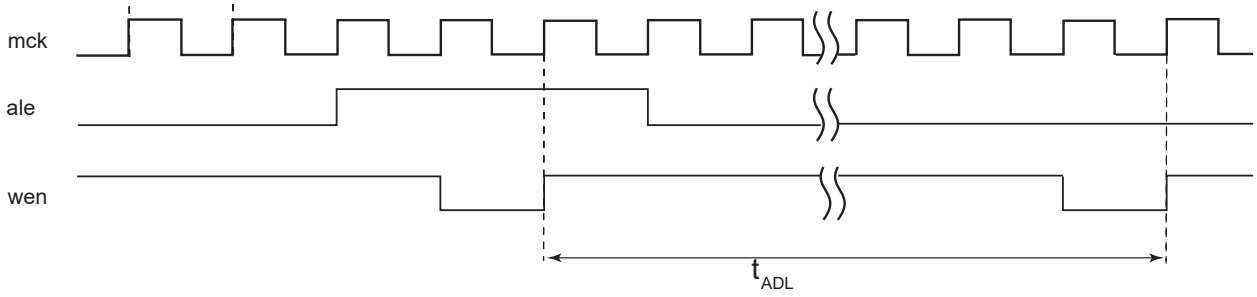



**Figure 36-33. Write Enable Timing for NAND Flash Device Data Input Mode**



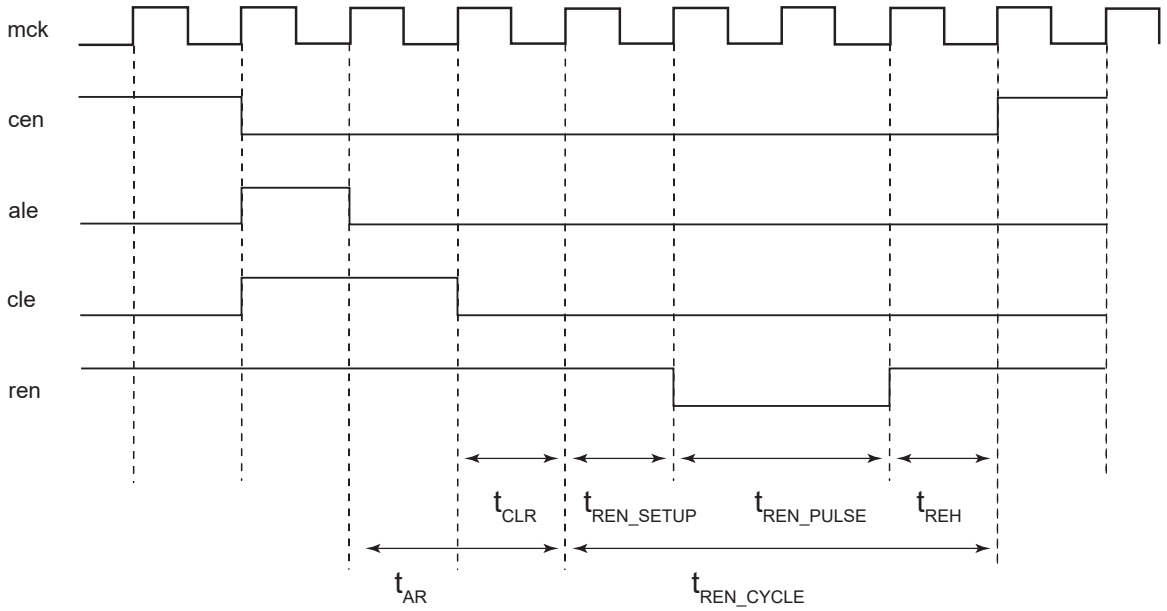

- Read Enable Configuration

Use NRD_SETUP, NRD_PULSE and NRD_CYCLE to define the read enable waveform according to the external device datasheet.

Use HSMC_TIMINGS.TAR to configure the timings between the address latch enable falling edge to read the enable falling edge.

Use HSMC_TIMINGS.TCLR to configure the timings between the command latch enable falling edge to read the enable falling edge.

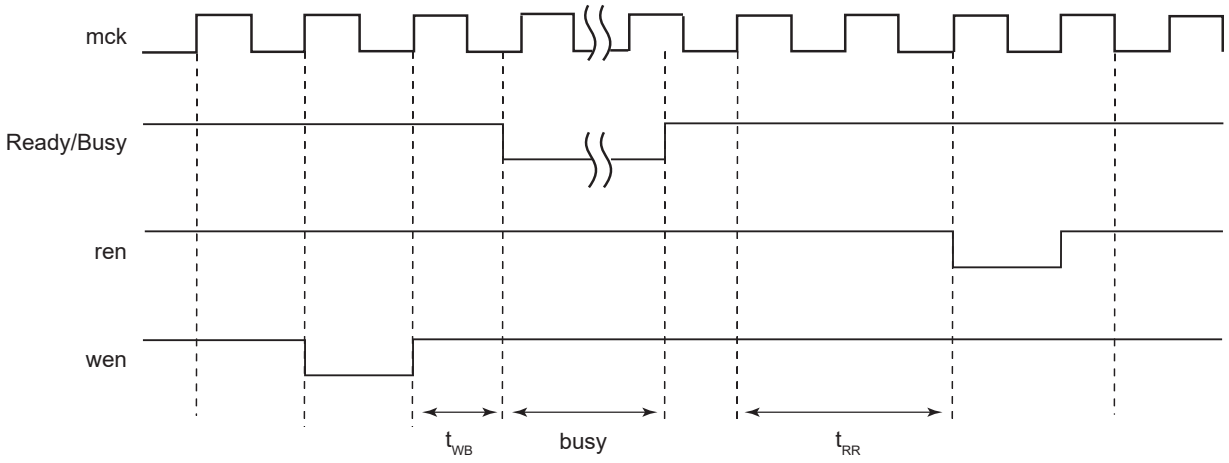
**Figure 36-34. Read Enable Timing Configuration Working with NAND Flash Device**




- Ready/Busy Signal Timing Configuration with a NAND Flash Device

Use HSMC_TIMINGS.TWB to configure the maximum elapsed time between the rising edge of the wen signal and the falling edge of the Ready/Busy signal. Use TRR field in the HSMC_TIMINGS register to program the number of clock cycles between the rising edge of the Ready/Busy signal and the falling edge of the ren signal.

**Figure 36-35. Ready/Busy Timing Configuration**

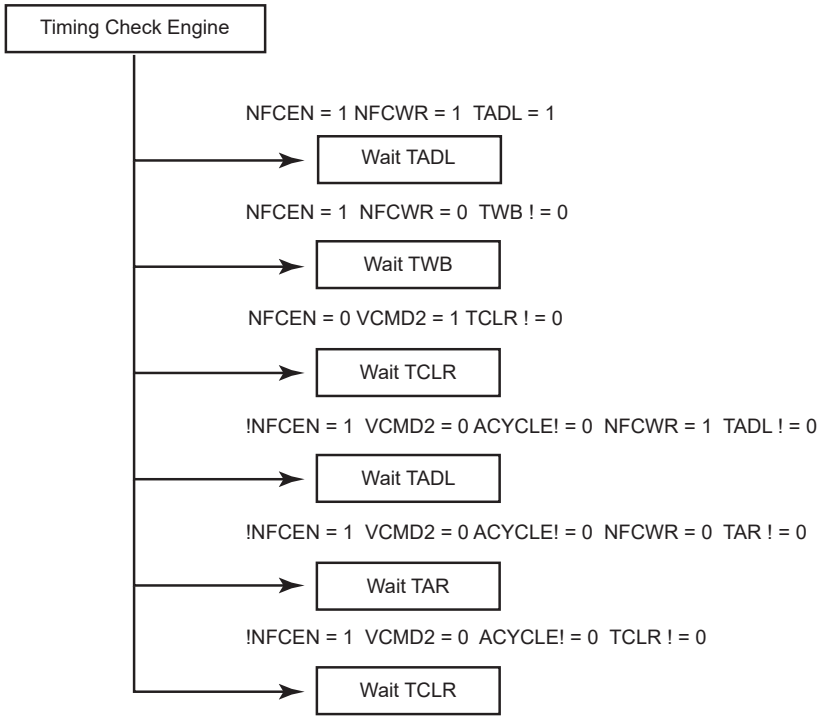



#### 36.17.3.1 NFC Timing Engine

When the NFC Command register is written, the NFC issues a NAND Flash Command and optionally performs a data transfer between the NFC SRAM and the NAND Flash device. The NFC Timing Engine guarantees valid NAND Flash timings, depending on the set of parameters decoded from the address bus. These timings are defined in the HSMC_TIMINGS register.

For information on the timing used depending on the command, see the figure below.

**Figure 36-36. NFC Timing Engine**




See the register descriptions 36.17.2.2 NFCADDR_CMD and 36.20.36 HSMC_TIMINGSx.

### 36.17.4 NFC SRAM

#### 36.17.4.1 NFC SRAM Mapping

If the NFC is used to read and write data from and to the NAND Flash, the configuration depends on the page size (PAGESIZE field in HSMC_CFG register). See the tables below for detailed mapping.

The NFC can handle the NAND Flash with a page size of 8 Kbytes or lower (such as 2 Kbytes, for example). In case of a 4 Kbyte or lower page size, the NFC SRAM can be split into two banks. The BANK bit in the HSMC_BANK register is used to select where NAND flash data are written or read. For an 8 Kbyte page size this field is not relevant.

Note that a “Ping-Pong” mode (write or read to a bank while the NFC writes or reads to another bank) is accessible with the NFC (using two different banks).

If the NFC is not used, the NFC SRAM can be used for a general purpose by the application.

**Table 36-8. NFC SRAM Bank Mapping for 512 bytes**

| Offset | Use | Access |
|---|---|---|
| 0x00000000–0x000001FF | Main Area Bank 0 | Read/Write |
| 0x00000200–0x000003FF | Spare Area Bank 0 | Read/Write |
| 0x00001200–0x000013FF | Main Area Bank 1 | Read/Write |
| 0x00001400–0x000015FF | Spare Area Bank 1 | Read/Write |

**Table 36-9. NFC SRAM Bank Mapping for 1 Kbyte**

| Offset | Use | Access |
|---|---|---|
| 0x00000000–0x000003FF | Main Area Bank 0 | Read/Write |
| 0x00000400–0x000005FF | Spare Area Bank 0 | Read/Write |
| 0x00001200–0x000015FF | Main Area Bank 1 | Read/Write |
| 0x00001600–0x000017FF | Spare Area Bank 1 | Read/Write |

**Table 36-10. NFC SRAM Bank Mapping for 2 Kbytes**

| Offset | Use | Access |
|---|---|---|
| 0x00000000–0x000007FF | Main Area Bank 0 | Read/Write |
| 0x00000800–0x000009FF | Spare Area Bank 0 | Read/Write |
| 0x00001200–0x000019FF | Main Area Bank 1 | Read/Write |
| 0x00001A00–0x00001BFF | Spare Area Bank 1 | Read/Write |

**Table 36-11. NFC SRAM Bank Mapping for 4 Kbytes**

| Offset | Use | Access |
|---|---|---|
| 0x00000000–0x00000FFF | Main Area Bank 0 | Read/Write |
| 0x00001000–0x000011FF | Spare Area Bank 0 | Read/Write |
| 0x00001200–0x000021FF | Main Area Bank 1 | Read/Write |
| 0x00002200–0x000023FF | Spare Area Bank 1 | Read/Write |

**Table 36-12. NFC SRAM Bank Mapping for 8 Kbytes, only one bank is available**

| Offset | Use | Access |
|---|---|---|
| 0x00000000–0x00001FFF | Main Area Bank 0 | Read/Write |
| 0x00002000–0x000023FF | Spare Area Bank 0 | Read/Write |

#### 36.17.4.2 NFC SRAM Access Prioritization Algorithm

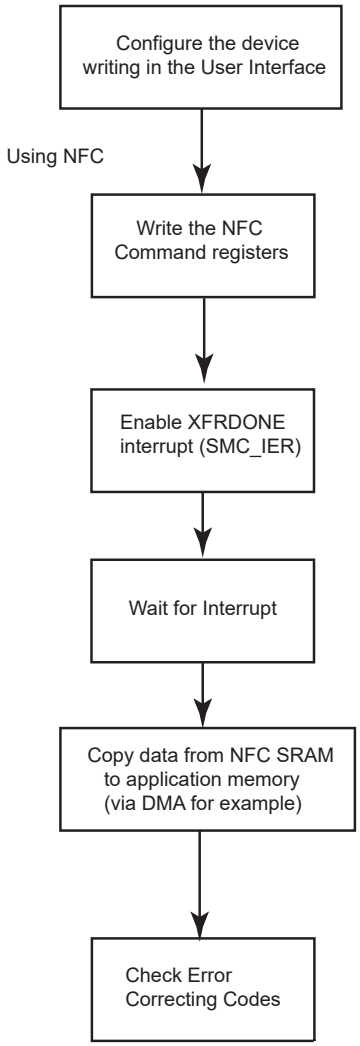
When the NFC is reading from or writing to an NFC SRAM bank, the other bank is available. If an NFC SRAM access occurs when the NFC performs a read or write operation in the same bank, then the access is discarded. The write operation is not performed. The read operation returns undefined data. If this situation is encountered, the AWB status flag located in the NFC Status Register is raised and indicates that a shared resource access violation has occurred.

### 36.17.5 NAND Flash Operations

This section describes the software operations needed to issue commands to the NAND Flash device and to perform data transfers using the NFC.

#### 36.17.5.1 Page Read

**Figure 36-37. Page Read Flow Chart**

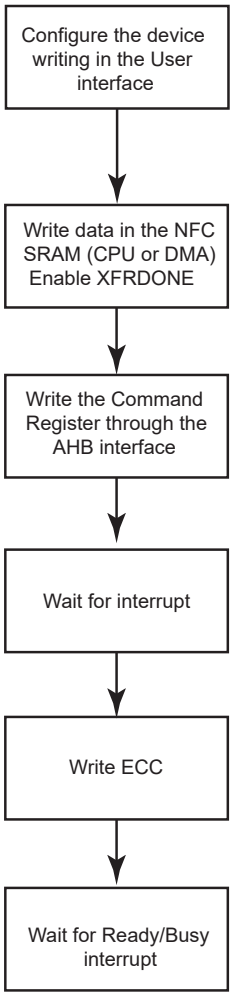



Instead of using the interrupt, the NFCBUSY flag can be polled.

For more information on the NFC Control register, see 36.17.2.2 NFCADDR_CMD.

#### 36.17.5.2 Program Page

**Figure 36-38. Program Page Flow Chart**




Writing the ECC cannot be done using the NFC; it needs to be done “manually”.

Instead of using the interrupt, the NFCBUSY flag can be polled.

For more information on the NFC Control register, see 36.17.2.2 NFCADDR_CMD .

## 36.18 PMECC Controller Functional Description

The Programmable Multibit Error Correcting Code (PMECC) controller is a programmable binary BCH (Bose, Chaudhuri and Hocquenghem) encoder/decoder. This controller can be used to generate redundancy information for both SLC and MLC NAND devices. It supports redundancy for correction of 2, 4, 8, 12, 24, or 32 errors per sector of data. The sector size is programmable and can be set to 512 bytes or 1024 bytes. The PMECC module generates redundancy at encoding time, when a NAND write page operation is performed. The redundancy is appended to the page and written in the spare area. This operation is performed by the processor. It moves the content of the PMECCX registers into the NAND flash memory. The number of registers depends on the selected error correction capability (see the table “Relevant Redundancy Registers”). This operation shall be executed for each sector. At decoding time, the PMECC module generates the remainders of the received codeword by the minimal polynomials.

When all remainders for a given sector are set to zero, no error occurred. When the remainders are different from zero, the codeword is corrupted and further processing is required.

The PMECC module generates an interrupt indicating that an error occurred. The processor must read the PMECC Interrupt Status Register (HSMC_PMECCISR). This register indicates which sector is corrupted.

The processor must execute the following decoding steps to find the error location within a sector:

1. Syndrome computation.
2. Finding the error location polynomial.
3. Finding the roots of the error location polynomial.

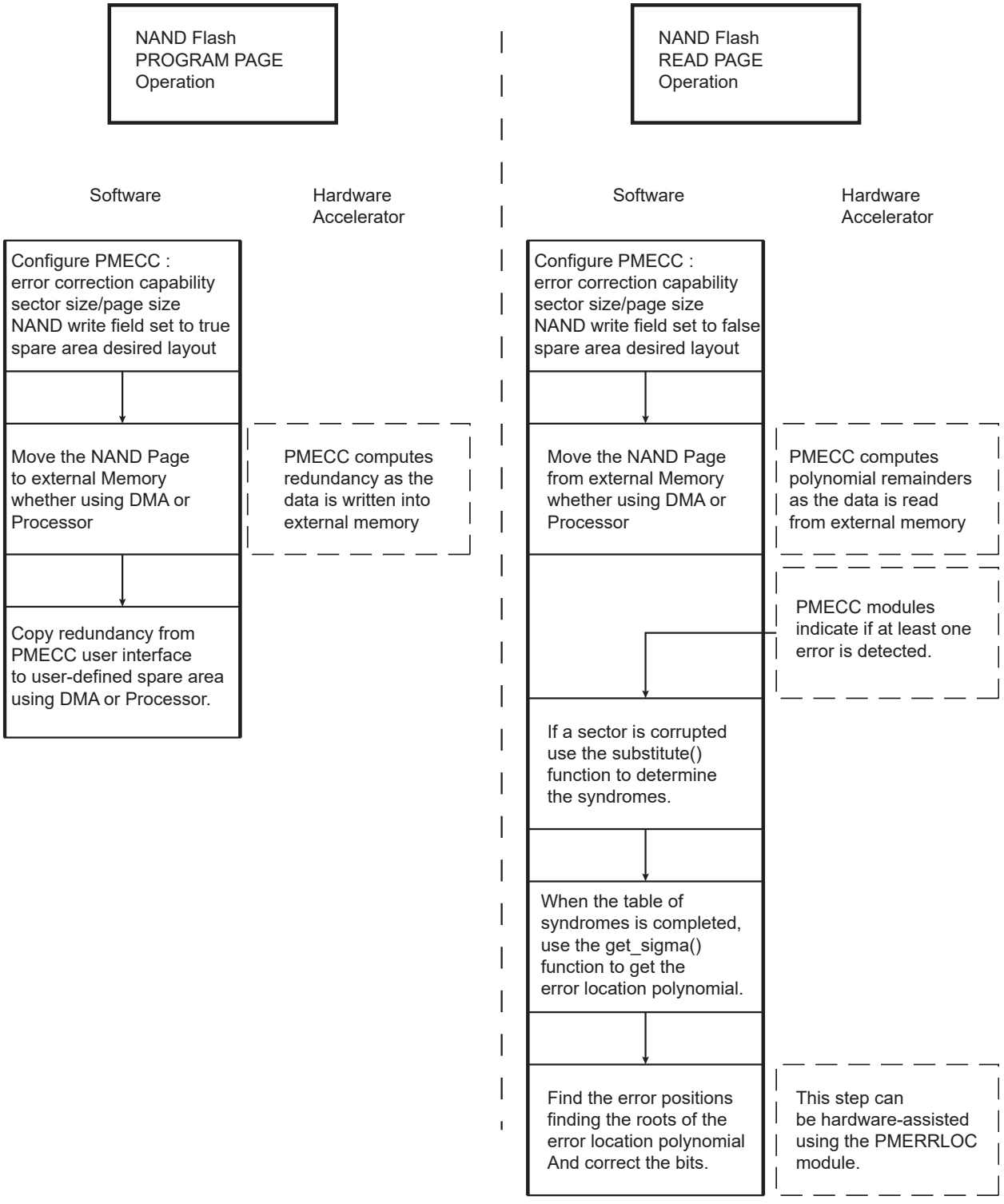
All decoding steps involve finite field computation. It means that a library of finite field arithmetic must be available to perform addition, multiplication and inversion. These arithmetic operations can be performed through the use of a memory mapped lookup table, or direct software implementation. The software implementation presented is based on lookup tables. Two tables named gf_log and gf_antilog are used. If alpha is the primitive element of the field, then a power of alpha is in the field. Assuming that beta = alpha ^ index, then beta belongs to the field, and gf_log(beta) = gf_log(alpha ^ index) = index. The gf_antilog table provides exponent inverse of the element; if beta = alpha ^ index, then gf_antilog(index) = beta.

The first step consists in the syndrome computation. The PMECC module computes the remainders and the software must substitute the power of the primitive element. The procedure implementation is given in the section “Remainder Substitution Procedure”.

The second step is the most software intensive. It is the Berlekamp’s iterative algorithm for finding the error-location polynomial. The procedure implementation is given in the section “Finding the Error Location Polynomial Sigma(x)”.

The Last step is finding the root of the error location polynomial. This step can be very software intensive. Indeed there is no straightforward method of finding the roots, except evaluating each element of the field in the error location polynomial. However, a hardware accelerator can be used to find the roots of the polynomial. The PMERRLOC module provides this kind of hardware acceleration.

**Figure 36-39. Software Hardware Multibit Error Correction Dataflow**




### 36.18.1 MLC/SLC Write Page Operation Using PMECC

When an MLC write page operation is performed, the PMECC controller is configured with the NANDWR bit of the PMECCFG register set to one. When the NAND spare area contains file system information and redundancy (PMECCx), the spare area is error protected, then the SPAREEN bit of the PMECCFG

register is set. When the NAND spare area contains only redundancy information, the SPAREEN bit is cleared.

When the write page operation is terminated, the user writes the redundancy in the NAND spare area. This operation can be done with DMA assistance.

**Table 36-13. Relevant Redundancy Registers**

| BCH_ERR Field | Sector Size Set to 512 Bytes | Sector Size Set to 1024 Bytes |
|---|---|---|
| 0 | PMECC0 | PMECC0 |
| 1 | PMECC0, PMECC1 | PMECC0, PMECC1 |
| 2 | PMECC0, PMECC1, PMECC2, PMECC3 | PMECC0, PMECC1, PMECC2, PMECC3 |
| 3 | PMECC0, PMECC1, PMECC2, PMECC3, PMECC4, PMECC5, PMECC6 | PMECC0, PMECC1, PMECC2, PMECC3, PMECC4, PMECC5, PMECC6 |
| 4 | PMECC0, PMECC1, PMECC2, PMECC3, PMECC4, PMECC5, PMECC6, PMECC7, PMECC8, PMECC9 | PMECC0, PMECC1, PMECC2, PMECC3, PMECC4, PMECC5, PMECC6, PMECC7, PMECC8, PMECC9, PMECC10 |
| 5 | PMECC0, PMECC1, PMECC2, PMECC3, PMECC4, PMECC5, PMECC6, PMECC7, PMECC8, PMECC9, PMECC10, PMECC11, PMECC12 | PMECC0, PMECC1, PMECC2, PMECC3, PMECC4, PMECC5, PMECC6, PMECC7, PMECC8, PMECC9, PMECC10, PMECC11, PMECC12, PMECC13 |

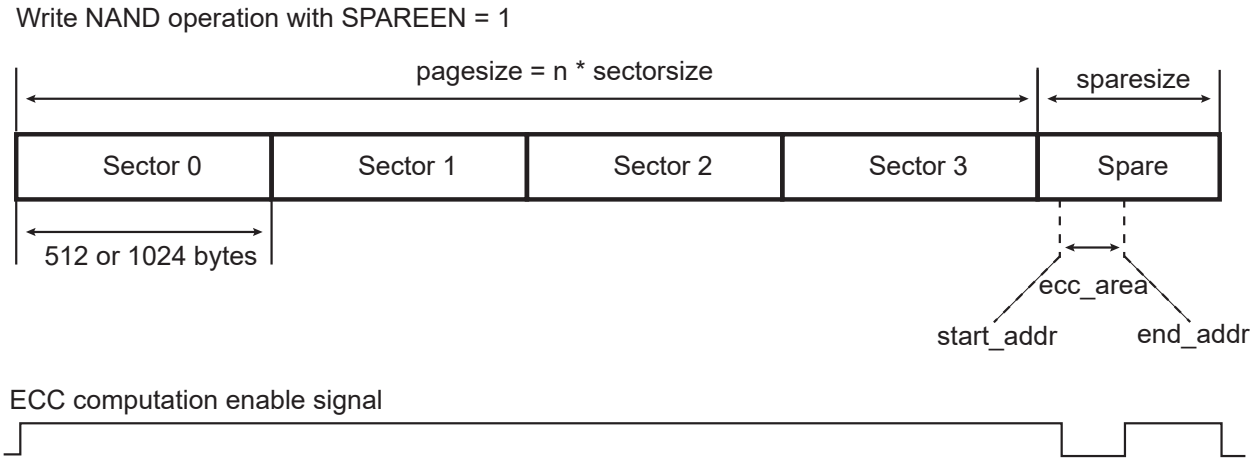
**Table 36-14. Number of Relevant ECC Bytes per Sector, Copied from LSByte to MSByte**

| BCH_ERR Field | Sector Size Set to 512 Bytes | Sector Size Set to 1024 Bytes |
|---|---|---|
| 0 | 4 bytes | 4 bytes |
| 1 | 7 bytes | 7 bytes |
| 2 | 13 bytes | 14 bytes |
| 3 | 20 bytes | 21 bytes |
| 4 | 39 bytes | 42 bytes |
| 5 | 52 bytes | 56 bytes |

#### 36.18.1.1 SLC/MLC Write Operation with Spare Enable Bit Set

When the SPAREEN bit of the PMECCFG register is set, the spare area of the page is encoded with the stream of data of the last sector of the page. This mode is entered by setting the DATA bit of the PMECCTRL register. When the encoding process is over, the redundancy shall be written to the spare area in User mode. The USER bit of the PMECCTRL register must be set.

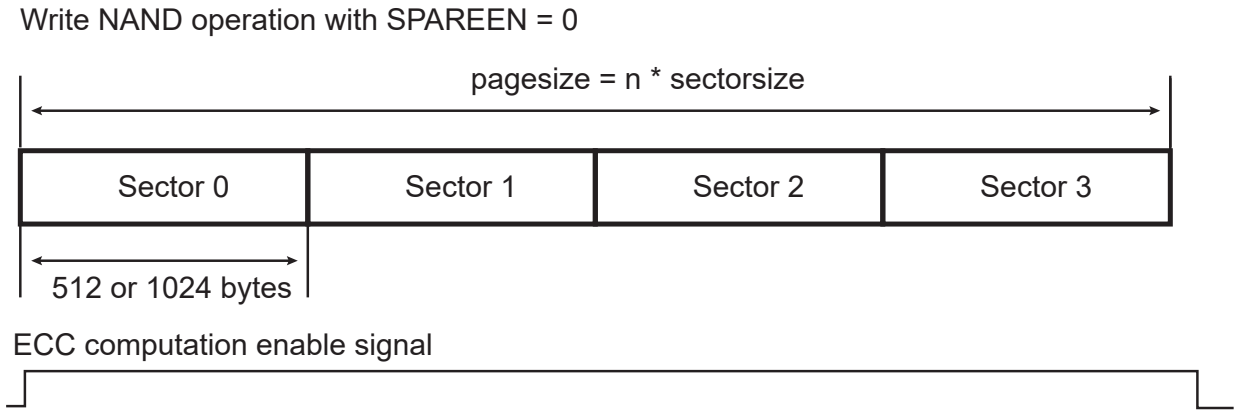
**Figure 36-40. NAND Write Operation with Spare Encoding**




### 36.18.1.2 SLC/MLC Write Operation with Spare Disable

When the SPAREEN bit of PMECCFG is cleared, the spare area is not encoded with the stream of data. This mode is entered by setting the DATA bit of the PMECCTRL register.

**Figure 36-41. NAND Write Operation**




### 36.18.2 MLC/SLC Read Page Operation Using PMECC

**Table 36-15. Relevant Remainder Registers**

| BCH_ERR Field | Sector Size Set to 512 Bytes | Sector Size Set to 1024 Bytes |
|---|---|---|
| 0 | PMECCREM0 | PMECCREM0 |
| 1 | PMECCREM0, PMECCREM1 | PMECCREM0, PMECCREM1 |
| 2 | PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3, | PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3 |
| 3 | PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3, PMECCREM4, PMECCREM5, PMECCREM6, PMECCREM7 | PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3, PMECCREM4, PMECCREM5, PMECCREM6, PMECCREM7 |

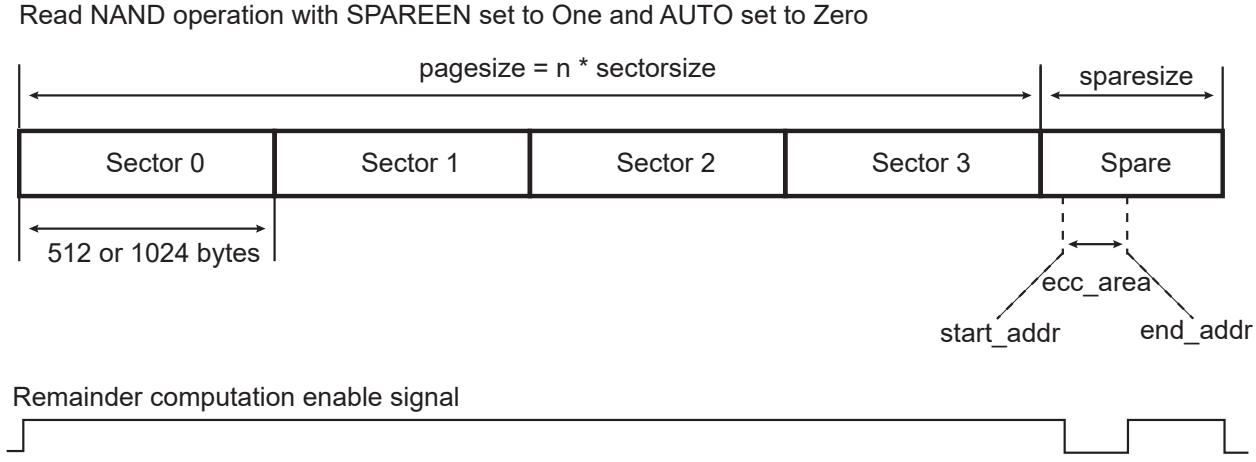
.....continued

| BCH_ERR Field | Sector Size Set to 512 Bytes | Sector Size Set to 1024 Bytes |
|---|---|---|
| 4 | PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3, PMECCREM4, PMECCREM5, PMECCREM6, PMECCREM7, PMECCREM8, PMECCREM9, PMECCREM10, PMECCREM11 | PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3, PMECCREM4, PMECCREM5, PMECCREM6, PMECCREM7, PMECCREM8, PMECCREM9, PMECCREM10, PMECCREM11 |
| 5 | PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3, PMECCREM4, PMECCREM5, PMECCREM6, PMECCREM7, PMECCREM8, PMECCREM9, PMECCREM10, PMECCREM11, PMECCREM12, PMECCREM13, PMECCREM14, PMECCREM15 | PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3, PMECCREM4, PMECCREM5, PMECCREM6, PMECCREM7, PMECCREM8, PMECCREM9, PMECCREM10, PMECCREM11, PMECCREM12, PMECCREM13, PMECCREM14, PMECCREM15 |

#### 36.18.2.1 MLC/SLC Read Operation with Spare Decoding

When the spare area is protected, it contains valid data. As the redundancy may be included in the middle of the information stream, the user shall program the start address and the end address of the ECC area. The controller will automatically skip the ECC area. This mode is entered writing a 1 in the DATA bit of the PMECCTRL register. When the page has been fully retrieved from the NAND, the ECC area shall be read using the User mode, writing a 1 to the USER bit of the PMECCTRL register.

**Figure 36-42. Read Operation with Spare Decoding**

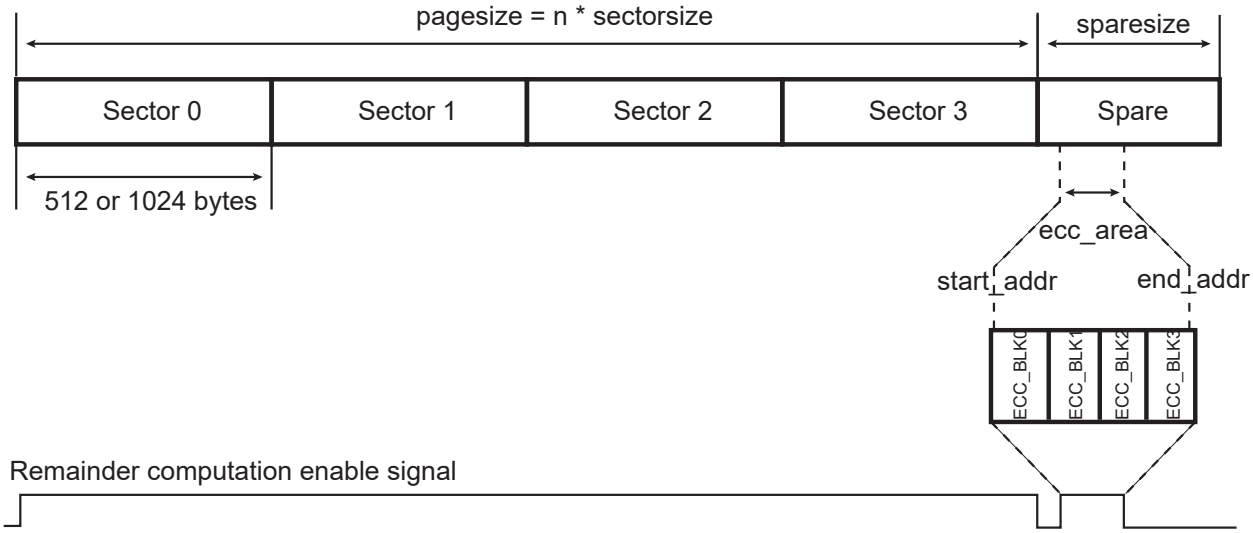



#### 36.18.2.2 MLC/SLC Read Operation

If the spare area is not protected with the error correcting code, the redundancy area is retrieved directly. This mode is entered writing a 1 in the DATA bit of the PMECCTRL register. When AUTO field is set to one, the ECC is retrieved automatically; otherwise, the ECC must be read using the User mode.

**Figure 36-43. Read Operation**

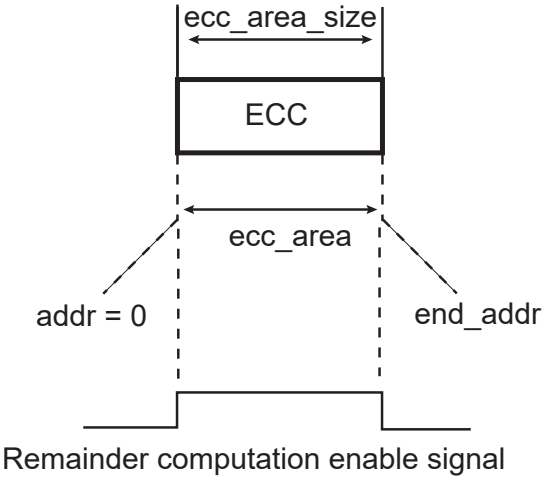
Read NAND operation with SPAREEN set to Zero and AUTO set to One




### 36.18.2.3 MLC/SLC User Read ECC Area

This mode allows a manual retrieve of the ECC. It is entered writing a 1 in the USER field of the PMECCTRL register.

**Figure 36-44. Read User Mode**




### 36.18.2.4 MLC Controller Working with NFC

**Table 36-16. MLC Controller Configuration when the Host Controller is Used**

| Transfer Type | NFC | | PMECC | | |
|---|---|---|---|---|---|
| | RSPARE | WSPARE | SPAREEN | AUTO | User Mode |
| Program Page main area is protected,<br>spare is not protected,<br>spare is written manually | 0 | 0 | 0 | 0 | Not used |

.....continued

| Transfer Type | NFC | | PMECC | | |
|---|---|---|---|---|---|
| | RSPARE | WSPARE | SPAREEN | AUTO | User Mode |
| Program Page main area is protected,<br>spare is protected,<br>spare is written by NFC | 0 | 1 | 1 | 0 | Not used |
| Read Page main area is protected,<br>spare is not protected,<br>spare is not retrieved by NFC | 0 | 0 | 0 | 0 | Used |
| Read Page main area is protected,<br>spare is not protected,<br>spare is retrieved by NFC | 1 | 0 | 0 | 1 | Not used |
| Read Page main area is protected,<br>spare is protected,<br>spare is retrieved by NFC | 1 | 0 | 1 | 0 | Used |

## 36.19 Software Implementation

### 36.19.1 Remainder Substitution Procedure

The substitute function evaluates the remainder polynomial, with different values of the field primitive element. The addition arithmetic operation is performed with the exclusive OR. The multiplication arithmetic operation is performed through the gf_log and gf_antilog lookup tables.

The REM2NP1 and REMN2NP3 fields of the PMECCREMN registers contain only odd remainders. Each bit indicates whether the coefficient of the remainder polynomial is set to zero or not.

NB_ERROR_MAX defines the maximum value of the error correcting capability.

NB_ERROR defines the error correcting capability selected at encoding/decoding time.

NB_FIELD_ELEMENTS defines the number of elements in the field.

si[] is a table that holds the current syndrome value. An element of that table belongs to the field. This is also a shared variable for the next step of the decoding operation.

oo[] is a table that contains the degree of the remainders.

```
int substitute()
{
int i;
int j;
for (i = 1; i < 2 * NB_ERROR_MAX; i++)
{
   si[i] = 0;
}
for (i = 1; i < 2*NB_ERROR; i++)
{
   for (j = 0; j < oo[i]; j++)
 {
     if (REM2NPX[i][j])
    {
      si[i] = gf_antilog[(i * j)%NB_FIELD_ELEMENTS] ^ si[i];
```

```
      }
    }
}
return 0;
}
```

### 36.19.2 Finding the Error Location Polynomial Sigma(x)

The sample code below gives a Berlekamp iterative procedure for finding the value of the error location polynomial.

The input of the procedure is the si[] table defined in the remainder substitution procedure.

The output of the procedure is the error location polynomial named smu (sigma mu). The polynomial coefficients belong to the field. The smu[NB_ERROR+1][] is a table that contains all these coefficients.

NB_ERROR_MAX defines the maximum value of the error correcting capability.

NB_ERROR defines the error correcting capability selected at encoding/decoding time.

NB_FIELD_ELEMENTS defines the number of elements in the field.

```
int get_sigma()
    {
    int i;
    int j;
    int k;
    /* mu          */
    int mu[NB_ERROR_MAX+2];
    /* sigma ro   */
    int sro[2*NB_ERROR_MAX+1];
    /* discrepancy */
    int dmu[NB_ERROR_MAX+2];
    /* delta order   */
    int delta[NB_ERROR_MAX+2];
    /* index of largest delta */
    int ro;
    int largest;
    int diff;
    /*                    */
    /*     First Row      */
    /*                    */
    /* Mu */
    mu[0]  = -1; /* Actually -1/2  */
    /* Sigma(x) set to 1  */
    for (i = 0; i < (2*NB_ERROR_MAX+1); i++)
     smu[0][i] = 0;
    smu[0][0] = 1;
    /* discrepancy set to 1 */
    dmu[0] = 1;
    /* polynom order set to 0 */
    lmu[0] = 0;
    /* delta set to -1 */
    delta[0]  = (mu[0] * 2 - lmu[0]) >> 1;
    /*                    */
    /*     Second Row     */
    /*                    */
    /* Mu */
    mu[1]  = 0;
    /* Sigma(x) set to 1 */
    for (i = 0; i < (2*NB_ERROR_MAX+1); i++)
     smu[1][i] = 0;
    smu[1][0] = 1;
    /* discrepancy set to Syndrome 1 */
    dmu[1] = si[1];
    /* polynom order set to 0 */
    lmu[1] = 0;
    /* delta set to 0 */
    delta[1]  = (mu[1] * 2 - lmu[1])>> 1;
    for (i=1; i <= NB_ERROR; i++)
    {
```

```
     mu[i+1] = i << 1;
    /*********************************************/
     /*                                           */
     /*                                           */
     /*          Compute Sigma (Mu+1)             */
     /*          And L(mu)                        */
     /* check if discrepancy is set to 0 */
     if (dmu[i] == 0)
     {
     /* copy polynom */
     for (j=0; j<2*NB_ERROR_MAX+1; j++)
     {
      smu[i+1][j] =
      smu[i][j];
     }
     /* copy previous polynom order to the next */
     lmu[i+1] = lmu[i];
     }
     else
     {
     ro      = 0;
     largest = -1;
     /* find largest delta with dmu != 0 */
     for (j=0; j<i; j++)
      {
      if (dmu[j])
       {
       if (delta[j] > largest)
        {
        largest = delta[j];
        ro   = j;
        }
       }
      }
     /* initialize signal ro */
     for (k = 0; k < 2*NB_ERROR_MAX+1; k ++)
     {
      sro[k] = 0;
     }
     /* compute difference */
     diff = (mu[i] - mu[ro]);
     /* compute X ^ (2(mu-ro)) */
     for (k = 0; k < (2*NB_ERROR_MAX+1); k ++)
     {
      sro[k+diff] =  smu[ro][k];
     }
     /* multiply by dmu * dmu[ro]^-1 */
     for (k = 0; k < 2*NB_ERROR_MAX+1; k ++)
     {
      /* dmu[ro] is not equal to zero by definition */
      /* check that operand are different from 0    */
      if (sro[k] && dmu[i])
      {
       /* galois inverse  */
       sro[k] = gf_antilog[(gf_log[dmu[i]] + (NB_FIELD_ELEMENTS-gf_log[dmu[ro]]) +
gf_log[sro[k]]) %
  NB_FIELD_ELEMENTS];
      }
     }
     /* multiply by dmu * dmu[ro]^-1 */
     for (k = 0; k < 2*NB_ERROR_MAX+1; k++)
     {
      smu[i+1][k] = smu[i][k] ^ sro[k];
      if (smu[i+1][k])
      {
       /* find the order of the polynom */
       lmu[i+1] = k << 1;
      }
     }
    }
    /*                                           */
    /*                                           */
    /*      End Compute Sigma (Mu+1)             */
    /*      And L(mu)                            */
    /*********************************************/
    /* In either case compute delta  */
```

```
delta[i+1]  = (mu[i+1] * 2 - lmu[i+1]) >> 1;
/* In either case compute the discrepancy */
for (k = 0 ; k <= (lmu[i+1]>>1); k++)
{
 if (k == 0)
 dmu[i+1] = si[2*(i-1)+3];
 /* check if one operand of the multiplier is null, its index is -1 */
 else if (smu[i+1][k] && si[2*(i-1)+3-k])
 dmu[i+1] = gf_antilog[(gf_log[smu[i+1][k]] + gf_log[si[2*(i-1)+3-k]])%nn] ^ dmu[i+1];
}
}
return 0;
}
```

### 36.19.3 Finding the Error Position

The output of the get_sigma() procedure is a polynomial stored in the smu[NB_ERROR+1][] table. The error positions are the roots of that polynomial. The degree of that polynomial is a very important information, as it gives the number of errors. PMERRLOC module provides hardware accelerator for that step.

#### 36.19.3.1 Error Location

The PMECC Error Location controller provides hardware acceleration for determining roots of polynomials over two finite fields: GF(2^13) and GF(2^14). It integrates 32 fully programmable coefficients. These coefficients belong to GF(2^13) or GF(2^14). The coefficient programmed in the PMERRLOC{i} is the coefficient of X ^ i in the polynomial.

The search operation is started as soon as a write access is detected in the ELEN register and can be disabled writing to the ELDIS register. The ENINIT field of the ELEN register shall be initialized with the number of galois field elements to test. The set of the roots can be limited to a valid range.

**Table 36-17. ENINIT Field Value for a Sector Size of 512 Bytes**

| Error Correcting Capability | ENINIT Value |
|---|---|
| 2 | 4122 |
| 4 | 4148 |
| 8 | 4200 |
| 12 | 4252 |
| 24 | 4408 |
| 32 | 4512 |

**Table 36-18. ENINIT Field Value for a Sector Size of 1024 Bytes**

| Error Correcting Capability | ENINIT Value |
|---|---|
| 2 | 8220 |
| 4 | 8248 |
| 8 | 8304 |
| 12 | 8360 |
| 24 | 8528 |
| 32 | 8640 |

When the PMECC engine is searching for roots, the BUSY field of the ELSR register remains asserted. An interrupt is asserted at the end of the computation, and the DONE bit of the PMECC Error Location Interrupt Status Register (HSMC_ELSIR) is set. The ERR_CNT field of the HSMC_ELISR indicates the number of errors. The error position can be read in the PMERRLOCX registers.

## 36.20 Register Summary

**Note:** The blocks of registers listed below are instanced 8 times in the user interface:

- HSMC_PMECCx[x=0..13]
- HSMCC_REMx[x=0..15]

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | HSMC_CFG | 7:0 | | | | | | PAGESIZE[2:0] | | |
| | | 15:8 | | | RBEDGE | EDGECTRL | | | RSPARE | WSPARE |
| | | 23:16 | | DTOMUL[2:0] | | | DTOCYC[3:0] | | | |
| | | 31:24 | | NFCSPARESIZE[6:0] | | | | | | |
| 0x04 | HSMC_CTRL | 7:0 | | | | | | | NFCDIS | NFCEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | HSMC_SR | 7:0 | | | RB_FALL | RB_RISE | | | | SMCSTS |
| | | 15:8 | | NFCSID[2:0] | | | NFCWR | | | NFCBUSY |
| | | 23:16 | NFCASE | AWB | UNDEF | DTOE | | | CMDDONE | XFRDONE |
| | | 31:24 | | | | | | | | RB_EDGE0 |
| 0x0C | HSMC_IER | 7:0 | | | RB_FALL | RB_RISE | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | NFCASE | AWB | UNDEF | DTOE | | | CMDDONE | XFRDONE |
| | | 31:24 | | | | | | | | RB_EDGE0 |
| 0x10 | HSMC_IDR | 7:0 | | | RB_FALL | RB_RISE | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | NFCASE | AWB | UNDEF | DTOE | | | CMDDONE | XFRDONE |
| | | 31:24 | | | | | | | | RB_EDGE0 |
| 0x14 | HSMC_IMR | 7:0 | | | RB_FALL | RB_RISE | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | NFCASE | AWB | UNDEF | DTOE | | | CMDDONE | XFRDONE |
| | | 31:24 | | | | | | | | RB_EDGE0 |
| 0x18 | HSMC_ADDR | 7:0 | ADDR_CYCLE0[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | HSMC_BANK | 7:0 | | | | | | | | BANK |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 ... 0x6F | Reserved | | | | | | | | | |
| 0x70 | HSMC_PMECCFG | 7:0 | | | | SECTORSZ | | BCH_ERR[2:0] | | |
| | | 15:8 | | | | NANDWR | | | PAGESIZE[1:0] | |
| | | 23:16 | | | | AUTO | | | | SPAREEN |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x74 | HSMC_PMECCSAREA | 7:0 | SPARESIZE[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | SPARESIZE[8:8] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x78 | HSMC_PMECCSADDR | 7:0 | STARTADDR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | STARTADDR[8:8] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x7C | HSMC_PMECCEADDR | 7:0 | ENDADDR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | ENDADDR[8:8] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x80 ... 0x83 | Reserved | | | | | | | | | |
| 0x84 | HSMC_PMECCTRL | 7:0 | | | DISABLE | ENABLE | | USER | DATA | RST |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x88 | HSMC_PMECCSR | 7:0 | | | | ENABLE | | | | BUSY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x8C | HSMC_PMECCIER | 7:0 | | | | | | | | ERRIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x90 | HSMC_PMECCIDR | 7:0 | | | | | | | | ERRID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x94 | HSMC_PMECCIMR | 7:0 | | | | | | | | ERRIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x98 | HSMC_PMECCISR | 7:0 | ERRIS[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x9C ... 0xAF | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0xB0 | HSMC_PMECC0 | 7:0 | ECC[7:0] | | | | | | | | |
| | | 15:8 | ECC[15:8] | | | | | | | | |
| | | 23:16 | ECC[23:16] | | | | | | | | |
| | | 31:24 | ECC[31:24] | | | | | | | | |
| ... | | | | | | | | | | | |
| 0xE4 | HSMC_PMECC13 | 7:0 | ECC[7:0] | | | | | | | | |
| | | 15:8 | ECC[15:8] | | | | | | | | |
| | | 23:16 | ECC[23:16] | | | | | | | | |
| | | 31:24 | ECC[31:24] | | | | | | | | |
| 0xE8 ... 0x02AF | Reserved | | | | | | | | | | |
| 0x02B0 | HSMC_REM0 | 7:0 | REM2NP1[7:0] | | | | | | | | |
| | | 15:8 | | | REM2NP1[13:8] | | | | | | |
| | | 23:16 | REM2NP3[7:0] | | | | | | | | |
| | | 31:24 | | | REM2NP3[13:8] | | | | | | |
| ... | | | | | | | | | | | |
| 0x02EC | HSMC_REM15 | 7:0 | REM2NP1[7:0] | | | | | | | | |
| | | 15:8 | | | REM2NP1[13:8] | | | | | | |
| | | 23:16 | REM2NP3[7:0] | | | | | | | | |
| | | 31:24 | | | REM2NP3[13:8] | | | | | | |
| 0x02F0 ... 0x04FF | Reserved | | | | | | | | | | |
| 0x0500 | HSMC_ELCFG | 7:0 | | | | | | | | | SECTORSZ |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | ERRNUM[4:0] | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x0504 | HSMC_ELPRIM | 7:0 | PRIMITIV[7:0] | | | | | | | | |
| | | 15:8 | PRIMITIV[15:8] | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x0508 | HSMC_ELEN | 7:0 | ENINIT[7:0] | | | | | | | | |
| | | 15:8 | | | ENINIT[13:8] | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x050C | HSMC_ELDIS | 7:0 | | | | | | | | | DIS |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x0510 | HSMC_ELSR | 7:0 | | | | | | | | | BUSY |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0514 | HSMC_ELIER | 7:0 | | | | | | | | DONE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0518 | HSMC_ELIDR | 7:0 | | | | | | | | DONE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x051C | HSMC_ELIMR | 7:0 | | | | | | | | DONE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0520 | HSMC_ELISR | 7:0 | | | | | | | | DONE |
| | | 15:8 | | | ERR_CNT[5:0] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0524 ... 0x0527 | Reserved | | | | | | | | | |
| 0x0528 | HSMC_SIGMA0 | 7:0 | SIGMA0[7:0] | | | | | | | |
| | | 15:8 | | | SIGMA0[13:8] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x052C | HSMC_SIGMA1 | 7:0 | SIGMAx[7:0] | | | | | | | |
| | | 15:8 | | | SIGMAx[13:8] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| ... | | | | | | | | | | |
| 0x05A8 | HSMC_SIGMA32 | 7:0 | SIGMAx[7:0] | | | | | | | |
| | | 15:8 | | | SIGMAx[13:8] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05AC | HSMC_ERRLOC0 | 7:0 | ERRLOCN[7:0] | | | | | | | |
| | | 15:8 | | | ERRLOCN[13:8] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| ... | | | | | | | | | | |
| 0x0628 | HSMC_ERRLOC31 | 7:0 | ERRLOCN[7:0] | | | | | | | |
| | | 15:8 | | | ERRLOCN[13:8] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x062C ... 0x06FF | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0700 | HSMC_SETUP0 | 7:0 | | | NWE_SETUP[5:0] | | | | | |
| | | 15:8 | | | NCS_WR_SETUP[5:0] | | | | | |
| | | 23:16 | | | NRD_SETUP[5:0] | | | | | |
| | | 31:24 | | | NCS_RD_SETUP[5:0] | | | | | |
| 0x0704 | HSMC_PULSE0 | 7:0 | | NWE_PULSE[6:0] | | | | | | |
| | | 15:8 | | NCS_WR_PULSE[6:0] | | | | | | |
| | | 23:16 | | NRD_PULSE[6:0] | | | | | | |
| | | 31:24 | | NCS_RD_PULSE[6:0] | | | | | | |
| 0x0708 | HSMC_CYCLE0 | 7:0 | NWE_CYCLE[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | NWE_CYCLE[8:8] |
| | | 23:16 | NRD_CYCLE[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | NRD_CYCLE[8:8] |
| 0x070C | HSMC_TIMINGS0 | 7:0 | TADL[3:0] | | | | TCLR[3:0] | | | |
| | | 15:8 | | | | OCMS | TAR[3:0] | | | |
| | | 23:16 | | | | | TRR[3:0] | | | |
| | | 31:24 | NFSEL | | | | TWB[3:0] | | | |
| 0x0710 | HSMC_MODE0 | 7:0 | | | EXNW_MODE[1:0] | | | | WRITE_MODE | READ_MODE |
| | | 15:8 | | | | DBW | | | | BAT |
| | | 23:16 | | | | TDF_MODE | TDF_CYCLES[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x0714 | HSMC_SETUP1 | 7:0 | | | NWE_SETUP[5:0] | | | | | |
| | | 15:8 | | | NCS_WR_SETUP[5:0] | | | | | |
| | | 23:16 | | | NRD_SETUP[5:0] | | | | | |
| | | 31:24 | | | NCS_RD_SETUP[5:0] | | | | | |
| 0x0718 | HSMC_PULSE1 | 7:0 | | NWE_PULSE[6:0] | | | | | | |
| | | 15:8 | | NCS_WR_PULSE[6:0] | | | | | | |
| | | 23:16 | | NRD_PULSE[6:0] | | | | | | |
| | | 31:24 | | NCS_RD_PULSE[6:0] | | | | | | |
| 0x071C | HSMC_CYCLE1 | 7:0 | NWE_CYCLE[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | NWE_CYCLE[8:8] |
| | | 23:16 | NRD_CYCLE[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | NRD_CYCLE[8:8] |
| 0x0720 | HSMC_TIMINGS1 | 7:0 | TADL[3:0] | | | | TCLR[3:0] | | | |
| | | 15:8 | | | | OCMS | TAR[3:0] | | | |
| | | 23:16 | | | | | TRR[3:0] | | | |
| | | 31:24 | NFSEL | | | | TWB[3:0] | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0724 | HSMC_MODE1 | 7:0 | | | EXNW_MODE[1:0] | | | | WRITE_MODE | READ_MODE |
| | | 15:8 | | | | DBW | | | | BAT |
| | | 23:16 | | | | TDF_MODE | TDF_CYCLES[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x0728 | HSMC_SETUP2 | 7:0 | | | NWE_SETUP[5:0] | | | | | |
| | | 15:8 | | | NCS_WR_SETUP[5:0] | | | | | |
| | | 23:16 | | | NRD_SETUP[5:0] | | | | | |
| | | 31:24 | | | NCS_RD_SETUP[5:0] | | | | | |
| 0x072C | HSMC_PULSE2 | 7:0 | | NWE_PULSE[6:0] | | | | | | |
| | | 15:8 | | NCS_WR_PULSE[6:0] | | | | | | |
| | | 23:16 | | NRD_PULSE[6:0] | | | | | | |
| | | 31:24 | | NCS_RD_PULSE[6:0] | | | | | | |
| 0x0730 | HSMC_CYCLE2 | 7:0 | NWE_CYCLE[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | NWE_CYCLE[8:8] |
| | | 23:16 | NRD_CYCLE[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | NRD_CYCLE[8:8] |
| 0x0734 | HSMC_TIMINGS2 | 7:0 | TADL[3:0] | | | | TCLR[3:0] | | | |
| | | 15:8 | | | | OCMS | TAR[3:0] | | | |
| | | 23:16 | | | | | TRR[3:0] | | | |
| | | 31:24 | NFSEL | | | | TWB[3:0] | | | |
| 0x0738 | HSMC_MODE2 | 7:0 | | | EXNW_MODE[1:0] | | | | WRITE_MODE | READ_MODE |
| | | 15:8 | | | | DBW | | | | BAT |
| | | 23:16 | | | | TDF_MODE | TDF_CYCLES[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x073C | HSMC_SETUP3 | 7:0 | | | NWE_SETUP[5:0] | | | | | |
| | | 15:8 | | | NCS_WR_SETUP[5:0] | | | | | |
| | | 23:16 | | | NRD_SETUP[5:0] | | | | | |
| | | 31:24 | | | NCS_RD_SETUP[5:0] | | | | | |
| 0x0740 | HSMC_PULSE3 | 7:0 | | NWE_PULSE[6:0] | | | | | | |
| | | 15:8 | | NCS_WR_PULSE[6:0] | | | | | | |
| | | 23:16 | | NRD_PULSE[6:0] | | | | | | |
| | | 31:24 | | NCS_RD_PULSE[6:0] | | | | | | |
| 0x0744 | HSMC_CYCLE3 | 7:0 | NWE_CYCLE[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | NWE_CYCLE[8:8] |
| | | 23:16 | NRD_CYCLE[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | NRD_CYCLE[8:8] |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0748 | HSMC_TIMINGS3 | 7:0 | TADL[3:0] | | | | TCLR[3:0] | | | |
| | | 15:8 | | | | OCMS | TAR[3:0] | | | |
| | | 23:16 | | | | | TRR[3:0] | | | |
| | | 31:24 | NFSEL | | | | TWB[3:0] | | | |
| 0x074C | HSMC_MODE3 | 7:0 | | | EXNW_MODE[1:0] | | | | WRITE_MODE | READ_MODE |
| | | 15:8 | | | | DBW | | | | BAT |
| | | 23:16 | | | | TDF_MODE | TDF_CYCLES[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x0750 ... 0x079F | Reserved | | | | | | | | | |
| 0x07A0 | HSMC_OCMS | 7:0 | | | | | | | SRSE | SMSE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x07A4 | HSMC_KEY1 | 7:0 | KEY1[7:0] | | | | | | | |
| | | 15:8 | KEY1[15:8] | | | | | | | |
| | | 23:16 | KEY1[23:16] | | | | | | | |
| | | 31:24 | KEY1[31:24] | | | | | | | |
| 0x07A8 | HSMC_KEY2 | 7:0 | KEY2[7:0] | | | | | | | |
| | | 15:8 | KEY2[15:8] | | | | | | | |
| | | 23:16 | KEY2[23:16] | | | | | | | |
| | | 31:24 | KEY2[31:24] | | | | | | | |
| 0x07AC ... 0x07E3 | Reserved | | | | | | | | | |
| 0x07E4 | HSMC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0x07E8 | HSMC_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |

### 36.20.1 NFC Configuration Register

**Name:** HSMC_CFG
**Offset:** 0x000
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | NFCSPARESIZE[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | DTOMUL[2:0] | | | DTOCYC[3:0] | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | RBEDGE | EDGECTRL | | | RSPARE | WSPARE |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PAGESIZE[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 30:24 – NFCSPARESIZE[6:0]** NAND Flash Spare Area Size Retrieved by the Host Controller
The spare size is set to (NFCSPARESIZE + 1) * 4 bytes. The spare area is only retrieved when RSPARE or WSPARE is activated.

**Bits 22:20 – DTOMUL[2:0]** Data Timeout Multiplier
These fields determine the maximum number of Master Clock cycles that the SMC waits until the detection of a rising edge on Ready/Busy signal.

If the data timeout set by DTOCYC and DTOMUL has been exceeded, the Data Timeout Error flag (DTOE) in the NFC Status Register (NFC_SR) raises.

Data Timeout Multiplier is defined by DTOMUL as shown in the following table:

| Value | Name | Description |
|---|---|---|
| 0 | X1 | DTOCYC |
| 1 | X16 | DTOCYC x 16 |
| 2 | X128 | DTOCYC x 128 |
| 3 | X256 | DTOCYC x 256 |
| 4 | X1024 | DTOCYC x 1024 |
| 5 | X4096 | DTOCYC x 4096 |
| 6 | X65536 | DTOCYC x 65536 |
| 7 | X1048576 | DTOCYC x 1048576 |

**Bits 19:16 – DTOCYC[3:0]** Data Timeout Cycle Number

**Bit 13 – RBEDGE** Ready/Busy Signal Edge Detection

| Value | Description |
|---|---|
| 0 | Indicates the level of the Ready/Busy line. |
| 1 | Iindicates that a transition has occurred on the Ready/Busy line. |

**Bit 12 – EDGECTRL** Rising/Falling Edge Detection Control

| Value | Description |
|---|---|
| 0 | Rising edge is detected. |
| 1 | Falling edge is detected. |

**Bit 9 – RSPARE** Read Spare Area

| Value | Description |
|---|---|
| 0 | The NFC skips the spare area in Read mode. |
| 1 | The NFC reads both main area and spare area in Read mode. |

**Bit 8 – WSPARE** Write Spare Area

| Value | Description |
|---|---|
| 0 | The NFC skips the spare area in Write mode. |
| 1 | The NFC writes both main area and spare area in Write mode. |

**Bits 2:0 – PAGESIZE[2:0]** Page Size of the NAND Flash Device

| Value | Name | Description |
|---|---|---|
| 0 | PS512 | Main area 512 bytes |
| 1 | PS1024 | Main area 1024 bytes |
| 2 | PS2048 | Main area 2048 bytes |
| 3 | PS4096 | Main area 4096 bytes |
| 4 | PS8192 | Main area 8192 bytes |

### 36.20.2 NFC Control Register

**Name:** HSMC_CTRL
**Offset:** 0x004
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | NFCDIS | NFCEN |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

**Bit 1 – NFCDIS** NAND Flash Controller Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the NAND Flash controller. |

**Bit 0 – NFCEN** NAND Flash Controller Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the NAND Flash controller. |

### 36.20.3 NFC Status Register

**Name:** HSMC_SR
**Offset:** 0x008
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | RB_EDGE0 |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFCASE | AWB | UNDEF | DTOE | | | CMDDONE | XFRDONE |
| Access | R | R | R | R | | | R | R |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | NFCSID[2:0] | | | NFCWR | | | NFCBUSY |
| Access | | R | R | R | R | | | R |
| Reset | | 0 | 0 | 0 | 0 | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | RB_FALL | RB_RISE | | | | SMCSTS |
| Access | | | R | R | | | | R |
| Reset | | | 0 | 0 | | | | 0 |

**Bit 24 – RB_EDGE0** Ready/Busy Line 0 Edge Detected
If set to one, this flag indicates that an edge has been detected on the Ready/Busy Line 0. Depending on HSMC_CFG.EDGECTRL, only rising or falling edge is detected. This flag is reset after a status read operation.

**Bit 23 – NFCASE** NFC Access Size Error
If set to one, this flag indicates that an illegal access has been detected in the NFC Memory Area. Only Word Access is allowed within the NFC memory area. This flag is reset after a status read operation.

**Bit 22 – AWB** Accessing While Busy
If set to one, this flag indicates that an AHB master has performed an access during the busy phase. This flag is reset after a status read operation.

**Bit 21 – UNDEF** Undefined Area Error
When set to one, this flag indicates that the processor performed an access in an undefined memory area. This flag is reset after a status read operation.

**Bit 20 – DTOE** Data Timeout Error
When set to one, this flag indicates that the Data timeout set be by DTOMUL and DTOCYC has been exceeded. This flag is reset after a status read operation.

**Bit 17 – CMDDONE** Command Done
When set to one, this flag indicates that the NFC has terminated the Command. This flag is reset after a status read operation.

**Bit 16 – XFRDONE** NFC Data Transfer Terminated
When set to one, this flag indicates that the NFC has terminated the Data Transfer. This flag is reset after a status read operation.

**Bits 14:12 – NFCSID[2:0]** NFC Chip Select ID (this field cannot be reset)
When a command is issued, this field indicates the value of the targeted chip select.

**Bit 11 – NFCWR** NFC Write/Read Operation (this field cannot be reset)
When a command is issued, this field indicates the current Read or Write Operation.

**Bit 8 – NFCBUSY** NFC Busy (this field cannot be reset)
When set to one, this flag indicates that the Controller is activated and accesses the memory device.

**Bit 5 – RB_FALL** Selected Ready Busy Falling Edge Detected
When set to one, this flag indicates that a falling edge on the Ready/Busy Line has been detected. This flag is reset after a status read operation. The Ready/Busy line is selected through the decoding of field HSMC_SR.NFCSID.

**Bit 4 – RB_RISE** Selected Ready Busy Rising Edge Detected
When set to one, this flag indicates that a rising edge on the Ready/Busy Line has been detected. This flag is reset after a status read operation. The Ready/Busy line is selected through the decoding of field HSMC_SR.NFCSID.

**Bit 0 – SMCSTS** NAND Flash Controller Status (this field cannot be reset)

| Value | Description |
|---|---|
| 0 | NAND Flash Controller disabled. |
| 1 | NAND Flash Controller enabled. |

### 36.20.4 NFC Interrupt Enable Register

**Name:** HSMC_IER
**Offset:** 0x00C
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | RB_EDGE0 |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFCASE | AWB | UNDEF | DTOE | | | CMDDONE | XFRDONE |
| Access | W | W | W | W | | | W | W |
| Reset | – | – | – | – | | | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | RB_FALL | RB_RISE | | | | |
| Access | | | W | W | | | | |
| Reset | | | – | – | | | | |

**Bit 24 – RB_EDGE0** Ready/Busy Line 0 Interrupt Enable

**Bit 23 – NFCASE** NFC Access Size Error Interrupt Enable

**Bit 22 – AWB** Accessing While Busy Interrupt Enable

**Bit 21 – UNDEF** Undefined Area Access Interrupt Enable

**Bit 20 – DTOE** Data Timeout Error Interrupt Enable

**Bit 17 – CMDDONE** Command Done Interrupt Enable

**Bit 16 – XFRDONE** Transfer Done Interrupt Enable

**Bit 5 – RB_FALL** Ready Busy Falling Edge Detection Interrupt Enable

**Bit 4 – RB_RISE** Ready Busy Rising Edge Detection Interrupt Enable

### 36.20.5 NFC Interrupt Disable Register

**Name:** HSMC_IDR
**Offset:** 0x010
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | RB_EDGE0 |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFCASE | AWB | UNDEF | DTOE | | | CMDDONE | XFRDONE |
| Access | W | W | W | W | | | W | W |
| Reset | – | – | – | – | | | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | RB_FALL | RB_RISE | | | | |
| Access | | | W | W | | | | |
| Reset | | | – | – | | | | |

**Bit 24 – RB_EDGE0** Ready/Busy Line 0 Interrupt Disable

**Bit 23 – NFCASE** NFC Access Size Error Interrupt Disable

**Bit 22 – AWB** Accessing While Busy Interrupt Disable

**Bit 21 – UNDEF** Undefined Area Access Interrupt Disable

**Bit 20 – DTOE** Data Timeout Error Interrupt Disable

**Bit 17 – CMDDONE** Command Done Interrupt Disable

**Bit 16 – XFRDONE** Transfer Done Interrupt Disable

**Bit 5 – RB_FALL** Ready Busy Falling Edge Detection Interrupt Disable

**Bit 4 – RB_RISE** Ready Busy Rising Edge Detection Interrupt Disable

### 36.20.6 NFC Interrupt Mask Register

**Name:** HSMC_IMR
**Offset:** 0x014
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:
0: Disables the corresponding interrupt.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | RB_EDGE0 |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFCASE | AWB | UNDEF | DTOE | | | CMDDONE | XFRDONE |
| Access | R | R | R | R | | | R | R |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | RB_FALL | RB_RISE | | | | |
| Access | | | R | R | | | | |
| Reset | | | 0 | 0 | | | | |

**Bit 24 – RB_EDGE0** Ready/Busy Line 0 Interrupt Mask

**Bit 23 – NFCASE** NFC Access Size Error Interrupt Mask

**Bit 22 – AWB** Accessing While Busy Interrupt Mask

**Bit 21 – UNDEF** Undefined Area Access Interrupt Mask5

**Bit 20 – DTOE** Data Timeout Error Interrupt Mask

**Bit 17 – CMDDONE** Command Done Interrupt Mask

**Bit 16 – XFRDONE** Transfer Done Interrupt Mask

**Bit 5 – RB_FALL** Ready Busy Falling Edge Detection Interrupt Mask

**Bit 4 – RB_RISE** Ready Busy Rising Edge Detection Interrupt Mask

© 2018 Microchip Technology Inc.

### 36.20.7 NFC Address Cycle Zero Register

**Name:** HSMC_ADDR
**Offset:** 0x018
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR_CYCLE0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – ADDR_CYCLE0[7:0]** NAND Flash Array Address Cycle 0
When five address cycles are used, ADDR_CYCLE0 is the first byte written to the NAND Flash (used by the NFC).

### 36.20.8 NFC Bank Register

**Name:** HSMC_BANK
**Offset:** 0x01C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | BANK |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – BANK** Bank Identifier

| Value | Description |
|---|---|
| 0 | Bank 0 is used. |
| 1 | Bank 1 is used. |

### 36.20.9 PMECC Configuration Register

**Name:** HSMC_PMECCFG
**Offset:** 0x070
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | AUTO | | | | SPAREEN |
| Access | | | | R/W | | | | R/W |
| Reset | | | | 0 | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | NANDWR | | | PAGESIZE[1:0] | |
| Access | | | | R/W | | | R/W | R/W |
| Reset | | | | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | SECTORSZ | | BCH_ERR[2:0] | | |
| Access | | | | R/W | | | | |
| Reset | | | | 0 | | 0 | 0 | 0 |

**Bit 20 – AUTO** Automatic Mode Enable
This bit is only relevant in NAND Read Mode, when spare enable is activated.

| Value | Description |
|---|---|
| 0 | Indicates that the spare area is not protected. In that case, the ECC computation takes into account the ECC area located in the spare area. (within the start address and the end address). |
| 1 | Indicates that the spare area is error protected. In this case, the ECC computation takes into account the whole spare area minus the ECC area in the ECC computation operation. |

**Bit 16 – SPAREEN** Spare Enable
For NAND write access:
0: The spare area is skipped

1: The spare area is protected with the last sector of data.

For NAND read access:

0: The spare area is skipped.

1: The spare area contains protected data or only redundancy information.

**Bit 12 – NANDWR** NAND Write Access

| Value | Description |
|---|---|
| 0 | NAND read access. |
| 1 | NAND write access. |

**Bits 9:8 – PAGESIZE[1:0]** Number of Sectors in the Page

| Value | Name | Description |
|---|---|---|
| 0 | PAGESIZE_1SEC | 1 sector for main area (512 or 1024 bytes). |
| 1 | PAGESIZE_2SEC | 2 sectors for main area (1024 or 2048 bytes). |
| 2 | PAGESIZE_4SEC | 4 sectors for main area (2048 or 4096 bytes). |
| 3 | PAGESIZE_8SEC | 8 sectors for main area (4096 or 8192 bytes). |

**Bit 4 – SECTORSZ** Sector Size

| Value | Description |
|---|---|
| 0 | The ECC computation is based on a sector of 512 bytes. |
| 1 | The ECC computation is based on a sector of 1024 bytes. |

**Bits 2:0 – BCH_ERR[2:0]** Error Correcting Capability

| Value | Name | Description |
|---|---|---|
| 0 | BCH_ERR2 | 2 errors |
| 1 | BCH_ERR4 | 4 errors |
| 2 | BCH_ERR8 | 8 errors |
| 3 | BCH_ERR12 | 12 errors |
| 4 | BCH_ERR24 | 24 errors |
| 5 | BCH_ERR32 | 32 errors |

### 36.20.10 PMECC Spare Area Size Register

**Name:** HSMC_PMECCSAREA
**Offset:** 0x074
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SPARESIZE[8:8] |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SPARESIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 8:0 – SPARESIZE[8:0]** Spare Area Size
Number of bytes in the spare area. The spare area size is equal to (SPARESIZE + 1) bytes.

### 36.20.11 PMECC Start Address Register

**Name:** HSMC_PMECCSADDR
**Offset:** 0x078
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | STARTADDR[8:8] |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | STARTADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 8:0 – STARTADDR[8:0]** ECC Area Start Address
This register is programmed with the start ECC start address. When STARTADDR is equal to 0, then the first ECC byte is located at the first byte of the spare area.

### 36.20.12 PMECC End Address Register

**Name:** HSMC_PMECCEADDR
**Offset:** 0x07C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ENDADDR[8:8] |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ENDADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 8:0 – ENDADDR[8:0]** ECC Area End Address
This register is programmed with the start ECC end address. When ENDADDR is equal to N, then the first ECC byte is located at byte N of the spare area.

### 36.20.13 PMECC Control Register

**Name:** HSMC_PMECCTRL
**Offset:** 0x084
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DISABLE | ENABLE | | USER | DATA | RST |
| Access | | | W | W | | W | W | W |
| Reset | | | – | – | | – | – | – |

**Bit 5 – DISABLE** PMECC Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the PMECC controller. |

**Bit 4 – ENABLE** PMECC Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the PMECC controller. |

**Bit 2 – USER** Start a User Mode Phase

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The PMECC controller enters a User mode phase. |

**Bit 1 – DATA** Start a Data Phase

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The PMECC controller enters a Data phase. |

**Bit 0 – RST** Reset the PMECC Module

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the PMECC controller. |

### 36.20.14 PMECC Status Register

**Name:** HSMC_PMECCSR
**Offset:** 0x088
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | ENABLE | | | | BUSY |
| Access | | | | R | | | | R |
| Reset | | | | 0 | | | | 0 |

**Bit 4 – ENABLE** PMECC Enable Bit

| Value | Description |
|---|---|
| 0 | PMECC controller disabled. |
| 1 | PMECC controller enabled. |

**Bit 0 – BUSY** PMECC Kernel is Busy

| Value | Description |
|---|---|
| 0 | PMECC controller finite state machine reached idle state. |
| 1 | PMECC controller finite state machine is processing the incoming byte stream. |

### 36.20.15 PMECC Interrupt Enable Register

**Name:** HSMC_PMECCIER
**Offset:** 0x08C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ERRIE |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – ERRIE** Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The Multibit error interrupt is enabled. An interrupt will be raised if at least one error is detected in at least one sector. |

### 36.20.16 PMECC Interrupt Disable Register

**Name:** HSMC_PMECCIDR
**Offset:** 0x090
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ERRID |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – ERRID** Error Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Multibit error interrupt disabled. |

### 36.20.17 PMECC Interrupt Mask Register

**Name:** HSMC_PMECCIMR
**Offset:** 0x094
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ERRIM |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 0 – ERRIM** Error Interrupt Mask

| Value | Description |
|---|---|
| 0 | Multibit error disabled. |
| 1 | Multibit error enabled. |

### 36.20.18 PMECC Interrupt Status Register

**Name:** HSMC_PMECCISR
**Offset:** 0x098
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERRIS[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – ERRIS[7:0]** Error Interrupt Status Register
When set to one, bit i of the HSMC_PMECCISR indicates that sector i is corrupted.

### 36.20.19 PMECC Redundancy x Register

**Name:** HSMC_PMECCx
**Offset:** 0xB0 + x*0x04 [x=0..13]
**Reset:** 0x00000000
**Property:** Read-only

**Note:** The block of registers HSMC_PMECCx[x=0..13] is instanced 8 times in the user interface.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ECC[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ECC[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ECC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ECC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ECC[31:0]** BCH Redundancy
This register contains the remainder of the division of the codeword by the generator polynomial.

### 36.20.20 PMECC Remainder x Register

**Name:** HSMC_REMx
**Offset:** 0x02B0 + x*0x04 [x=0..15]
**Reset:** 0x00000000
**Property:** Read-only

**Note:** The block of registers HSMC_REMx[x=0..15] is instanced 8 times in the user interface.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | REM2NP3[13:8] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | REM2NP3[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | REM2NP1[13:8] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | REM2NP1[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:16 – REM2NP3[13:0]** BCH Remainder 2 * N + 3
When sector size is set to 512 bytes, bit REM2NP3[29] is not used and read as zero.

If bit i of the REM2NP3 field is set to one, then the coefficient of the $X^i$ is set to one; otherwise, the coefficient is zero.

**Bits 13:0 – REM2NP1[13:0]** BCH Remainder 2 * N + 1
When sector size is set to 512 bytes, bit REM2NP1[13] is not used and read as zero.

If bit i of the REM2NP1 field is set to one, then the coefficient of the $X^i$ is set to one; otherwise, the coefficient is zero.

### 36.20.21 PMECC Error Location Configuration Register

**Name:** HSMC_ELCFG
**Offset:** 0x500
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | ERRNUM[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SECTORSZ |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 20:16 – ERRNUM[4:0]** Number of Errors

**Bit 0 – SECTORSZ** Sector Size

| Value | Description |
|---|---|
| 0 | The ECC computation is based on a 512-byte sector. |
| 1 | The ECC computation is based on a 1024-byte sector. |

### 36.20.22 PMECC Error Location Primitive Register

**Name:** HSMC_ELPRIM
**Offset:** 0x504
**Reset:** 0x401A
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PRIMITIV[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PRIMITIV[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |

**Bits 15:0 – PRIMITIV[15:0]** Primitive Polynomial
This field indicates the Primitive Polynomial used in the ECC computation.

### 36.20.23 PMECC Error Location Enable Register

**Name:** HSMC_ELEN
**Offset:** 0x508
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | ENINIT[13:8] | | | | | |
| Access | | | W | W | W | W | W | W |
| Reset | | | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ENINIT[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 13:0 – ENINIT[13:0]** Error Location Enable
Initial bit number in the codeword.

### 36.20.24 PMECC Error Location Disable Register

**Name:** HSMC_ELDIS
**Offset:** 0x50C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

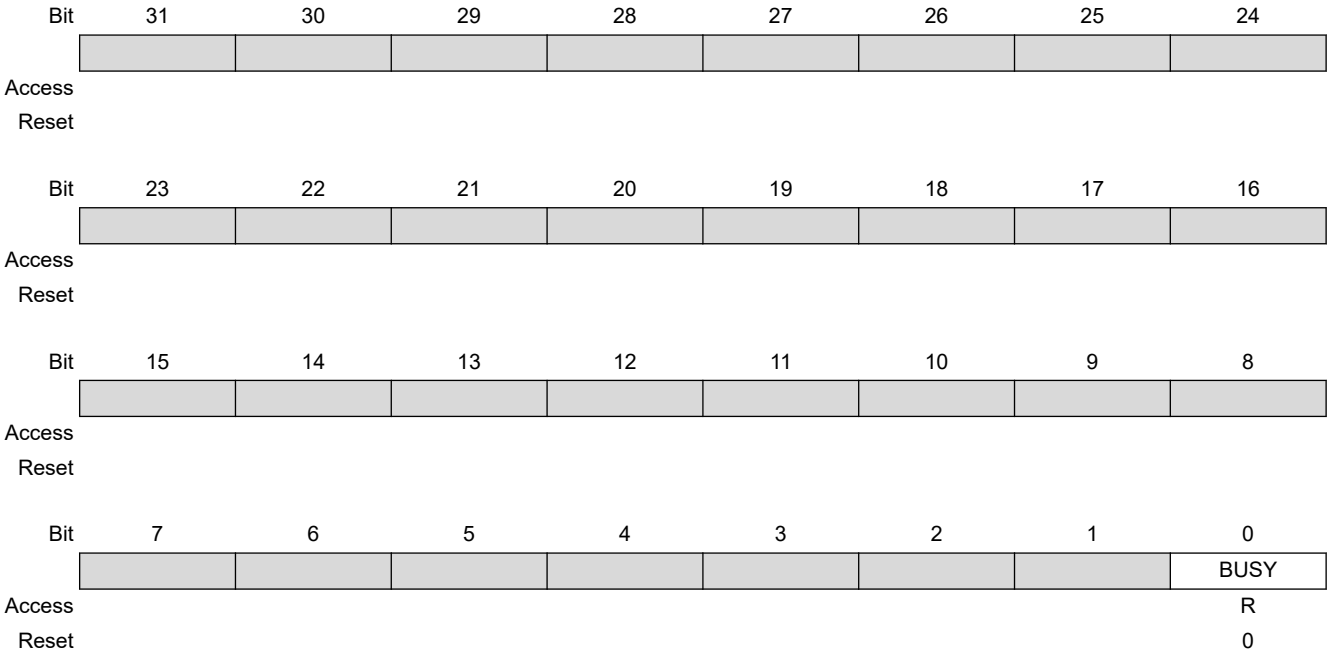
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DIS |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – DIS** Disable Error Location Engine

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Error location engine. |

### 36.20.25 PMECC Error Location Status Register

**Name:** HSMC_ELSR
**Offset:** 0x510
**Reset:** 0x00000000
**Property:** Read-only




**Bit 0 – BUSY** Error Location Engine Busy

| Value | Description |
|---|---|
| 0 | Error location engine is disabled. |
| 1 | Error location engine is enabled and is finding roots of the polynomial. |

### 36.20.26 PMECC Error Location Interrupt Enable Register

**Name:** HSMC_ELIER
**Offset:** 0x514
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DONE |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – DONE** Computation Terminated Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the interrupt. |

### 36.20.27 PMECC Error Location Interrupt Disable Register

**Name:** HSMC_ELIDR
**Offset:** 0x518
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DONE |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – DONE** Computation Terminated Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the interrupt. |

### 36.20.28 PMECC Error Location Interrupt Mask Register

**Name:** HSMC_ELIMR
**Offset:** 0x51C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DONE |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 0 – DONE** Computation Terminated Interrupt Mask

| Value | Description |
|---|---|
| 0 | Computation Terminated interrupt disabled. |
| 1 | Computation Terminated interrupt enabled. |

### 36.20.29 PMECC Error Location Interrupt Status Register

**Name:** HSMC_ELISR
**Offset:** 0x520
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | ERR_CNT[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DONE |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 13:8 – ERR_CNT[5:0]** Error Counter value
This field indicates the number of roots of the polynomial.

**Bit 0 – DONE** Computation Terminated Interrupt Status
When set to one, this indicates that the error location engine has completed the root finding algorithm.

### 36.20.30 PMECC Error Location SIGMA0 Register

**Name:** HSMC_SIGMA0
**Offset:** 0x528
**Reset:** 0x00000001
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | SIGMA0[13:8] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SIGMA0[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**Bits 13:0 – SIGMA0[13:0]** Coefficient of degree 0 in the SIGMA polynomial
SIGMA0 belongs to the finite field GF($2^{13}$) when the sector size is set to 512 bytes.

SIGMA0 belongs to the finite field GF($2^{14}$) when the sector size is set to 1024 bytes.

### 36.20.31 PMECC Error Location SIGMAx Register

**Name:** HSMC_SIGMAx
**Offset:** 0x052C + x*0x04 [x=0..31]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | SIGMAx[13:8] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SIGMAx[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 13:0 – SIGMAx[13:0]** Coefficient of degree x in the SIGMA polynomial
SIGMAx belongs to the finite field GF($2^{13}$) when the sector size is set to 512 bytes.

SIGMAx belongs to the finite field GF($2^{14}$) when the sector size is set to 1024 bytes.

### 36.20.32 PMECC Error Location x Register

**Name:** HSMC_ERRLOCx
**Offset:** 0x05AC + x*0x04 [x=0..31]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | ERRLOCN[13:8] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERRLOCN[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 13:0 – ERRLOCN[13:0]** Error Position within the Set {sector area, spare area}
ERRLOCN points to 1 when the first bit of the main area is corrupted.

If the sector size is set to 512 bytes, the ERRLOCN points to 4096 when the last bit of the sector area is corrupted.

If the sector size is set to 1024 bytes, the ERRLOCN points to 8192 when the last bit of the sector area is corrupted.

If the sector size is set to 512 bytes, the ERRLOCN points to 4097 when the first bit of the spare area is corrupted.

If the sector size is set to 1024 bytes, the ERRLOCN points to 8193 when the first bit of the spare area is corrupted.

### 36.20.33 Setup Register

**Name:** HSMC_SETUPx
**Offset:** 0x0700 + x*0x14 [x=0..3]
**Reset:** 0x01010101
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | NCS_RD_SETUP[5:0] | | | | | |
| Access | | | W | W | W | W | W | W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | NRD_SETUP[5:0] | | | | | |
| Access | | | W | W | W | W | W | W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | NCS_WR_SETUP[5:0] | | | | | |
| Access | | | W | W | W | W | W | W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | NWE_SETUP[5:0] | | | | | |
| Access | | | W | W | W | W | W | W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 1 |

**Bits 29:24 – NCS_RD_SETUP[5:0]** NCS Setup Length in Read Access
In Read access, the NCS signal setup length is defined as:

NCS setup length = (128 * NCS_RD_SETUP[5] + NCS_RD_SETUP[4:0]) clock cycles.

**Bits 21:16 – NRD_SETUP[5:0]** NRD Setup Length
The NRD signal setup length is defined as:

NRD setup length = (128 * NRD_SETUP[5] + NRD_SETUP[4:0]) clock cycles.

**Bits 13:8 – NCS_WR_SETUP[5:0]** NCS Setup Length in Write Access
In write access, the NCS signal setup length is defined as:

NCS setup length = (128 * NCS_WR_SETUP[5] + NCS_WR_SETUP[4:0]) clock cycles.

**Bits 5:0 – NWE_SETUP[5:0]** NWE Setup Length
The NWE signal setup length is defined as:

NWE setup length = (128 * NWE_SETUP[5] + NWE_SETUP[4:0]) clock cycles.

### 36.20.34 Pulse Register

**Name:** HSMC_PULSEx
**Offset:** 0x0704 + x*0x14 [x=0..3]
**Reset:** 0x01010101
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | NCS_RD_PULSE[6:0] | | | | | | |
| Access | | W | W | W | W | W | W | W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | NRD_PULSE[6:0] | | | | | | |
| Access | | W | W | W | W | W | W | W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | NCS_WR_PULSE[6:0] | | | | | | |
| Access | | W | W | W | W | W | W | W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | NWE_PULSE[6:0] | | | | | | |
| Access | | W | W | W | W | W | W | W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**Bits 30:24 – NCS_RD_PULSE[6:0]** NCS Pulse Length in READ Access
In READ mode, The NCS signal pulse length is defined as:

NCS pulse length = (256 * NCS_RD_PULSE[6] + NCS_RD_PULSE[5:0]) clock cycles.

**Bits 22:16 – NRD_PULSE[6:0]** NRD Pulse Length
The NRD signal pulse length is defined as:

NRD pulse length = (256 * NRD_PULSE[6] + NRD_PULSE[5:0]) clock cycles.

The NRD pulse width must be as least 1 clock cycle.

**Bits 14:8 – NCS_WR_PULSE[6:0]** NCS Pulse Length in WRITE Access
In Write access, The NCS signal pulse length is defined as:

NCS pulse length = (256 * NCS_WR_PULSE[6] + NCS_WR_PULSE[5:0]) clock cycles.

The NCS pulse must be at least one clock cycle.

**Bits 6:0 – NWE_PULSE[6:0]** NWE Pulse Length
The NWE signal pulse length is defined as:

NWE pulse length = (256 * NWE_PULSE[6]+NWE_PULSE[5:0]) clock cycles.

The NWE pulse must be at least one clock cycle.

### 36.20.35 Cycle Register

**Name:** HSMC_CYCLEx
**Offset:** 0x0708 + x*0x14 [x=0..3]
**Reset:** 0x00030003
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | NRD_CYCLE[8:8] |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NRD_CYCLE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | NWE_CYCLE[8:8] |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NWE_CYCLE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

**Bits 24:16 – NRD_CYCLE[8:0]** Total Read Cycle Length
The total read cycle length is the total duration in clock cycles of the read cycle. It is equal to the sum of the setup, pulse and hold steps of the NRD and NCS signals. It is defined as:

Read cycle length = (NRD_CYCLE[8:7] * 256) + NRD_CYCLE[6:0] clock cycles.

**Bits 8:0 – NWE_CYCLE[8:0]** Total Write Cycle Length
The total write cycle length is the total duration in clock cycles of the write cycle. It is equal to the sum of the setup, pulse and hold steps of the NWE and NCS signals. It is defined as:

Write cycle length = (NWE_CYCLE[8:7] * 256) + NWE_CYCLE[6:0] clock cycles.

### 36.20.36 Timings Register

**Name:** HSMC_TIMINGSx
**Offset:** 0x070C + x*0x14 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFSEL | | | | TWB[3:0] | | | |
| Access | R/W | | | | R/W | R/W | R/W | R/W |
| Reset | 0 | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TRR[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | OCMS | TAR[3:0] | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TADL[3:0] | | | | TCLR[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – NFSEL** NAND Flash Selection
If this bit is set to one, the chip select is assigned to NAND Flash write enable and read enable lines drive the Error Correcting Code module.

**Bits 27:24 – TWB[3:0]** WEN High to REN to Busy
Write Enable rising edge to Ready/Busy falling edge timing.

Write Enable to Read/Busy = (TWB[3] * 64) + TWB[2:0] clock cycles.

**Bits 19:16 – TRR[3:0]** Ready to REN Low Delay
Ready/Busy signal to Read Enable falling edge timing.

Read to REN = (TRR[3] * 64) + TRR[2:0] clock cycles.

**Bit 12 – OCMS** Off Chip Memory Scrambling Enable
When set to one, the memory scrambling is activated. (Value must be zero if external memory is NAND Flash and NFC is used).

**Bits 11:8 – TAR[3:0]** ALE to REN Low Delay
Address Latch Enable falling edge to Read Enable falling edge timing.

Address Latch Enable to Read Enable = (TAR[3] * 64) + TAR[2:0] clock cycles.

**Bits 7:4 – TADL[3:0]** ALE to Data Start
Last address latch cycle to the first rising edge of WEN for data input.

Last address latch to first rising edge of WEN = (TADL[3] * 64) + TADL[2:0] clock cycles.

**Bits 3:0 – TCLR[3:0]** CLE to REN Low Delay
Command Latch Enable falling edge to Read Enable falling edge timing.

Latch Enable Falling to Read Enable Falling = (TCLR[3] * 64) + TCLR[2:0] clock cycles.

### 36.20.37 Mode Register

**Name:** HSMC_MODEx
**Offset:** 0x0710 + x*0x14 [x=0..3]
**Reset:** 0x00001003
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | TDF_MODE | TDF_CYCLES[3:0] | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | DBW | | | | BAT |
| Access | | | | R/W | | | | R/W |
| Reset | | | | 1 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | EXNW_MODE[1:0] | | | | WRITE_MODE | READ_MODE |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 1 | 1 |

**Bit 20 – TDF_MODE** TDF Optimization

| Value | Description |
|---|---|
| 0 | TDF optimization disabled<br>– The number of TDF wait states is inserted before the next access begins. |
| 1 | TDF optimization enabled<br>– The number of TDF wait states is optimized using the setup period of the next read/write access. |

**Bits 19:16 – TDF_CYCLES[3:0]** Data Float Time
This field gives the integer number of clock cycles required by the external device to release the data after the rising edge of the read controlling signal. The SMC always provide one full cycle of bus turnaround after the TDF_CYCLES period. The external bus cannot be used by another chip select during TDF_CYCLES + 1 cycles. From 0 up to 15 TDF_CYCLES can be set.

**Bit 12 – DBW** Data Bus Width

| Value | Name | Description |
|---|---|---|
| 0 | BIT_8 | 8-bit bus |
| 1 | BIT_16 | 16-bit bus |

**Bit 8 – BAT** Byte Access Type
This field is used only if DBW defines a 16-bit data bus.

| Value | Name | Description |
|---|---|---|
| 0 | BYTE_SELECT | Byte select access type:<br>- Write operation is controlled using NCS, NWE, NBS0, NBS1.<br>- Read operation is controlled using NCS, NRD, NBS0, NBS1. |
| 1 | BYTE_WRITE | Byte write access type:<br>- Write operation is controlled using NCS, NWR0, NWR1.<br>- Read operation is controlled using NCS and NRD. |

**Bits 5:4 – EXNW_MODE[1:0]** NWAIT Mode
The NWAIT signal is used to extend the current read or write signal. It is only taken into account during the pulse phase Read and Write controlling signal. When the use of NWAIT is enabled, at least one cycle hold duration must be programmed for the read and write controlling signal.

| Value | Name | Description |
|---|---|---|
| 0 | DISABLED | Disabled—The NWAIT input signal is ignored on the corresponding Chip Select. |
| 1 | – | Reserved |
| 2 | FROZEN | Frozen Mode—If asserted, the NWAIT signal freezes the current read or write cycle. After deassertion, the read/write cycle is resumed from the point where it was stopped. |
| 3 | READY | Ready Mode—The NWAIT signal indicates the availability of the external device at the end of the pulse of the controlling read or write signal, to complete the access. If high, the access normally completes. If low, the access is extended until NWAIT returns high. |

**Bit 1 – WRITE_MODE** Selection of the Control Signal for Write Operation

| Value | Name | Description |
|---|---|---|
| 0 | NCS_CTRL | The Write operation is controller by the NCS signal. |
| 1 | NWE_CTRL | The Write operation is controlled by the NWE signal |

**Bit 0 – READ_MODE** Selection of the Control Signal for Read Operation

| Value | Name | Description |
|---|---|---|
| 0 | NCS_CTRL | The Read operation is controlled by the NCS signal. |
| 1 | NRD_CTRL | The Read operation is controlled by the NRD signal. |

### 36.20.38 Off Chip Memory Scrambling Register

**Name:** HSMC_OCMS
**Offset:** 0x7A0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | SRSE | SMSE |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – SRSE** NFC Internal SRAM Scrambling Enable

| Value | Description |
|---|---|
| 0 | Disable Scrambling for NFC internal SRAM access. |
| 1 | Enable Scrambling for NFC internal SRAM access. (OCMS bit must be cleared in the corresponding HSMC_TIMINGSx register.) |

**Bit 0 – SMSE** Static Memory Controller Scrambling Enable

| Value | Description |
|---|---|
| 0 | Disable “Off Chip” Scrambling for SMC access. |
| 1 | Enable “Off Chip” Scrambling for SMC access. (If OCMS bit is set in the corresponding HSMC_TIMINGSx register.) |

### 36.20.39 Off Chip Memory Scrambling Key1 Register

**Name:** HSMC_KEY1
**Offset:** 0x7A4
**Reset:** 0x00000000
**Property:** Write-once

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY1[31:24] | | | | | | | |
| Access | W-Once | W-Once | W-Once | W-Once | W-Once | W-Once | W-Once | W-Once |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEY1[23:16] | | | | | | | |
| Access | W-Once | W-Once | W-Once | W-Once | W-Once | W-Once | W-Once | W-Once |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | KEY1[15:8] | | | | | | | |
| Access | W-Once | W-Once | W-Once | W-Once | W-Once | W-Once | W-Once | W-Once |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | KEY1[7:0] | | | | | | | |
| Access | W-Once | W-Once | W-Once | W-Once | W-Once | W-Once | W-Once | W-Once |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – KEY1[31:0]** Off Chip Memory Scrambling (OCMS) Key Part 1
When Off Chip Memory Scrambling is enabled by setting the HSMC_OCMS and HSMC_TIMINGS registers in accordance, the data scrambling depends on KEY1 and KEY2 values.

### 36.20.40 Off Chip Memory Scrambling Key2 Register

**Name:** HSMC_KEY2
**Offset:** 0x7A8
**Reset:** 0x00000000
**Property:** Write-once

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY2[31:24] | | | | | | | |
| Access | W-once | W-once | W-once | W-once | W-once | W-once | W-once | W-once |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEY2[23:16] | | | | | | | |
| Access | W-once | W-once | W-once | W-once | W-once | W-once | W-once | W-once |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | KEY2[15:8] | | | | | | | |
| Access | W-once | W-once | W-once | W-once | W-once | W-once | W-once | W-once |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | KEY2[7:0] | | | | | | | |
| Access | W-once | W-once | W-once | W-once | W-once | W-once | W-once | W-once |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – KEY2[31:0]** Off Chip Memory Scrambling (OCMS) Key Part 2
When Off Chip Memory Scrambling is enabled by setting the HSMC_OCMS and HSMC_TIMINGS registers in accordance, the data scrambling depends on KEY2 and KEY1 values.

### 36.20.41 Write Protection Mode Register

**Name:** HSMC_WPMR
**Offset:** 0x7E4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x534D43 | PASSWD | Writing any other value in this field aborts the write operation of bit WPEN. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See Section 1.16 “Register Write Protection” for list of write-protected registers.

| Value | Description |
|---|---|
| 0 | Disables write protection if WPKEY value corresponds to 0x534D43 (“SMC” in ASCII) |
| 1 | Enables write protection if WPKEY value corresponds to 0x534D43 (“SMC” in ASCII) |

### 36.20.42 Write Protection Status Register

**Name:** HSMC_WPSR
**Offset:** 0x7E8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protect violation has occurred since the last read of the HSMC_WPSR. |
| 1 | A write protect violation has occurred since the last read of the HSMC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 37. DMA Controller (XDMAC)

## 37.1 Description

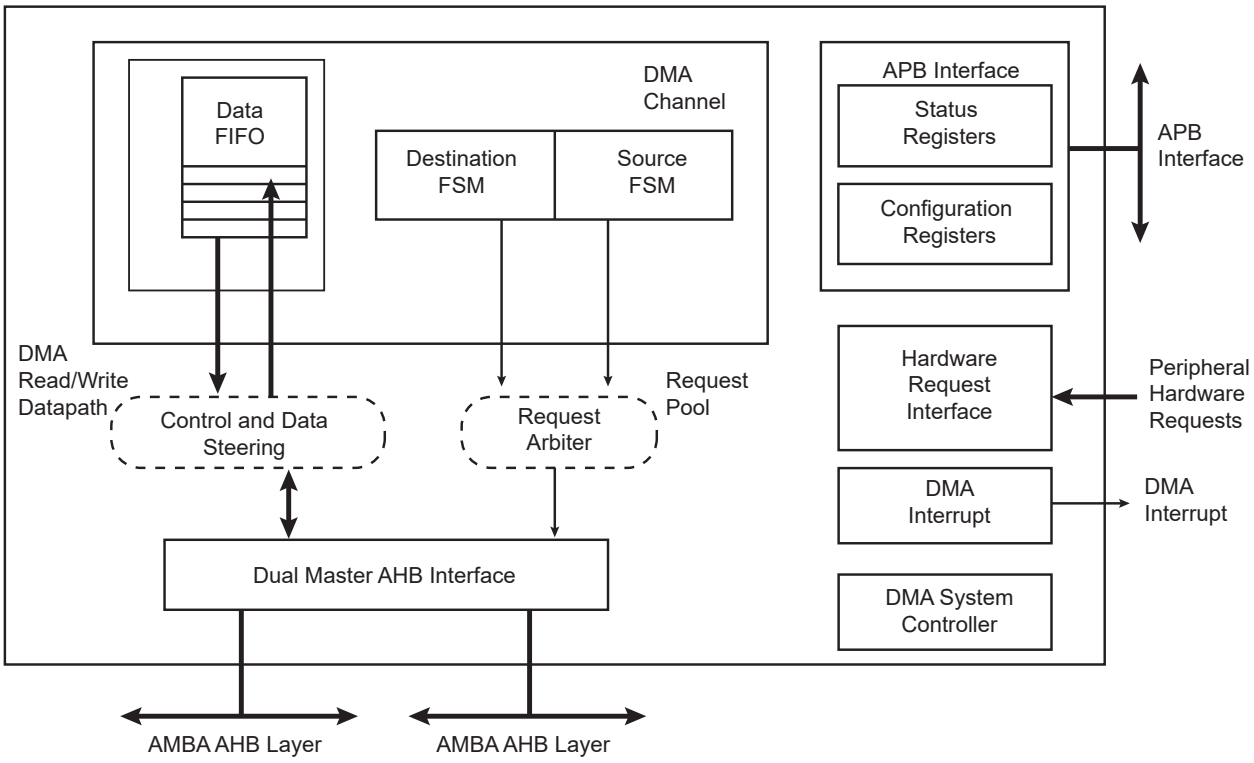
The DMA Controller (XDMAC) is a AHB-protocol central direct memory access controller. It performs peripheral data transfer and memory move operations over one or two bus ports through the unidirectional communication channel. Each channel is fully programmable and provides both peripheral or memory-to-memory transfers. The channel features are configurable at implementation.

## 37.2 Embedded Characteristics

- 2 Master Interfaces
- 16 DMA Channels
- 51 Hardware Requests
- 4 Kbytes Embedded FIFO
- Supports Peripheral-to-Memory, Memory-to-Peripheral, or Memory-to-Memory Transfer Operations
- Peripheral DMA Operation Runs on Bytes (8-bit), Half-Word (16-bit) and Word (32-bit)
- Memory DMA Operation Runs on Bytes (8 bit), Half-Word (16-bit), Word (32-bit) and Double-Word (64-bit)
- Supports Hardware and Software Initiated Transfers
- Supports Linked List Operations
- Supports Incrementing or Fixed Addressing Mode
- Supports Programmable Independent Data Striding for Source and Destination
- Supports Programmable Independent Microblock Striding for Source and Destination
- Configurable Priority Group and Arbitration Policy
- Programmable AHB Burst Length
- Configuration Interface Accessible through APB Interface
- XDMAC Architecture Includes Multiport FIFO
- Supports Multiple View Channel Descriptor
- Automatic Flush of Channel Trailing Bytes
- Automatic Coarse-Grain and Fine-Grain Clock Gating
- Hardware Acceleration of Memset Pattern

## 37.3 Block Diagram

**Figure 37-1. DMA Controller (XDMAC) Block Diagram**




## 37.4 DMA Controller Peripheral Connections

The SAMA5D2 features two DMACs: XDMAC0 and XDMAC1. Both have the same features:

- Programmable secure access
- Two 64-bit masters
- 16 channels and 55 hardware requests embedded
- Sixteen 64-bit-word FIFOs on all channels
- Linked list support with status write back operation at end of transfer
- Word, half-word, byte transfer support
- Memory-to-memory transfer
- Peripheral-to-memory transfer
- Memory-to-peripheral transfer

The DMA controller can handle the transfer between peripherals and memory and so receives the triggers from the peripherals below.

The following table gives an overview of the different access when secure/non-secure DMA needs to access a secure/non-secure peripheral, and when a secure/non-secure peripheral needs to access secure/non-secure DMA.

**Table 37-1. DMA Configuration vs. Peripheral**

| Peripheral | DMA | |
|---|---|---|
| | Secure | Non-secure |
| Secure | x | – |
| Non-secure | x | x |

DMA Controller 0 manages transfers between peripherals and memory, and receives the triggers from the peripherals listed ine following table.

**Table 37-2. DMA Channels Definition (XDMAC0)**

| Instance Name | Channel T/R | Interface Number | XDMAC_CCx.CSIZE Required Value |
|---|---|---|---|
| TWIHS0 | Transmit | 0 | 0 |
| TWIHS0 | Receive | 1 | |
| TWIHS1 | Transmit | 2 | 0 |
| TWIHS1 | Receive | 3 | |
| QSPI0 | Transmit | 4 | 0 |
| QSPI0 | Receive | 5 | |
| SPI0 | Transmit | 6 | 0 |
| SPI0 | Receive | 7 | |
| SPI1 | Transmit | 8 | 0 |
| SPI1 | Receive | 9 | |
| PWM | Transmit | 10 | 0 |
| FLEXCOM0 | Transmit | 11 | 0 |
| FLEXCOM0 | Receive | 12 | |
| FLEXCOM1 | Transmit | 13 | 0 |
| FLEXCOM1 | Receive | 14 | |
| FLEXCOM2 | Transmit | 15 | 0 |
| FLEXCOM2 | Receive | 16 | |
| FLEXCOM3 | Transmit | 17 | 0 |
| FLEXCOM3 | Receive | 18 | |
| FLEXCOM4 | Transmit | 19 | 0 |
| FLEXCOM4 | Receive | 20 | |
| SSC0 | Transmit | 21 | 0 |
| SSC0 | Receive | 22 | |

.....continued

| Instance Name | Channel T/R | Interface Number | XDMAC_CCx.CSIZE Required Value |
|---|---|---|---|
| SSC1 | Transmit | 23 | 0 |
| SSC1 | Receive | 24 | |
| ADC | Receive | 25 | 0 |
| AES | Transmit | 26 | 0 or 2 (refer to chapter AES, section Start Mode, subsection DMA Mode) |
| AES | Receive | 27 | |
| TDES | Transmit | 28 | 0 |
| TDES | Receive | 29 | |
| SHA | Transmit | 30 | 4 |
| I2SC0 | Transmit | 31 | 0 |
| I2SC0 | Receive | 32 | |
| I2SC1 | Transmit | 33 | 0 |
| I2SC1 | Receive | 34 | |
| UART0 | Transmit | 35 | 0 |
| UART0 | Receive | 36 | |
| UART1 | Transmit | 37 | 0 |
| UART1 | Receive | 38 | |
| UART2 | Transmit | 39 | 0 |
| UART2 | Receive | 40 | |
| UART3 | Transmit | 41 | 0 |
| UART3 | Receive | 42 | |
| UART4 | Transmit | 43 | 0 |
| UART4 | Receive | 44 | |
| TC0 | Receive | 45 | 0 |
| TC1 | Receive | 46 | |
| CLASSD | Transmit | 47 | 0 |
| QSPI1 | Transmit | 48 | 0 |
| QSPI1 | Receive | 49 | |
| PDMIC | Receive | 50 | 0 |

DMA Controller 1 manages transfers between peripherals and memory, and receives the triggers from the peripherals listed in the following table.

**Table 37-3. DMA Channels Definition (XDMAC1)**

| Instance Name | Channel T/R | Interface Number | XDMAC_CCx.CSIZE Required Value |
|---|---|---|---|
| TWIHS0 | Transmit | 0 | 0 |
| TWIHS0 | Receive | 1 | |
| TWIHS1 | Transmit | 2 | 0 |
| TWIHS1 | Receive | 3 | |
| QSPI0 | Transmit | 4 | 0 |
| QSPI0 | Receive | 5 | |
| SPI0 | Transmit | 6 | 0 |
| SPI0 | Receive | 7 | |
| SPI1 | Transmit | 8 | 0 |
| SPI1 | Receive | 9 | |
| PWM | Transmit | 10 | 0 |
| FLEXCOM0 | Transmit | 11 | 0 |
| FLEXCOM0 | Receive | 12 | |
| FLEXCOM1 | Transmit | 13 | 0 |
| FLEXCOM1 | Receive | 14 | |
| FLEXCOM2 | Transmit | 15 | 0 |
| FLEXCOM2 | Receive | 16 | |
| FLEXCOM3 | Transmit | 17 | 0 |
| FLEXCOM3 | Receive | 18 | |
| FLEXCOM4 | Transmit | 19 | 0 |
| FLEXCOM4 | Receive | 20 | |
| SSC0 | Transmit | 21 | 0 |
| SSC0 | Receive | 22 | |
| SSC1 | Transmit | 23 | 0 |
| SSC1 | Receive | 24 | |
| ADC | Receive | 25 | 0 |
| AES | Transmit | 26 | 0 or 2 (refer to chapter AES, section Start Mode, subsection DMA Mode) |
| AES | Receive | 27 | |
| TDES | Transmit | 28 | 0 |
| TDES | Receive | 29 | |
| SHA | Transmit | 30 | 4 |

.....continued

| Instance Name | Channel T/R | Interface Number | XDMAC_CCx.CSIZE Required Value |
|---|---|---|---|
| I2SC0 | Transmit | 31 | 0 |
| I2SC0 | Receive | 32 | |
| I2SC1 | Transmit | 33 | 0 |
| I2SC1 | Receive | 34 | |
| UART0 | Transmit | 35 | 0 |
| UART0 | Receive | 36 | |
| UART1 | Transmit | 37 | 0 |
| UART1 | Receive | 38 | |
| UART2 | Transmit | 39 | 0 |
| UART2 | Receive | 40 | |
| UART3 | Transmit | 41 | 0 |
| UART3 | Receive | 42 | |
| UART4 | Transmit | 43 | 0 |
| UART4 | Receive | 44 | |
| TC0 | Receive | 45 | 0 |
| TC1 | Receive | 46 | |
| CLASSD | Transmit | 47 | 0 |
| QSPI1 | Transmit | 48 | 0 |
| QSPI1 | Receive | 49 | |
| PDMIC | Receive | 50 | 0 |

## 37.5 Functional Description

### 37.5.1 Basic Definitions

**Source Peripheral**: Slave device, memory mapped on the interconnection network, from where the XDMAC reads data. The source peripheral teams up with a destination peripheral to form a channel. A data read operation is scheduled when the peripheral transfer request is asserted.

**Destination Peripheral**: Slave device, memory mapped on the interconnection network, to which the XDMAC writes. A write data operation is scheduled when the peripheral transfer request is asserted.

**Channel**: The data movement between source and destination creates a logical channel.

**Stride**: Number of address locations between successive elements/data measured in bytes.

**Transfer Type**: The transfer is hardware-synchronized when it is paced by the peripheral hardware request, otherwise the transfer is self-triggered (memory to memory transfer).

**XDMAC Master Transfer**: The Master Transfer is composed of a linked list of blocks. The channel address, control and configuration registers can be modified at the inter block boundary. The descriptor structure modifies the channel registers conditionally. Interrupts can be generated on a per block basis or when the end of linked list event occurs.

**XDMAC Block**: An XDMAC block is composed of a programmable number of microblocks. The channel configuration registers remain unchanged at the inter microblock boundary. The source and destination addresses are conditionally updated with a programmable signed number.

**XDMAC Microblock**: The microblock is composed of a programmable number of data. The channel configuration registers remain unchanged at the data boundary. The data address may be fixed (a FIFO location, a peripheral transmit or receive register), incrementing (a memory-mapped area) by a programmable signed number.

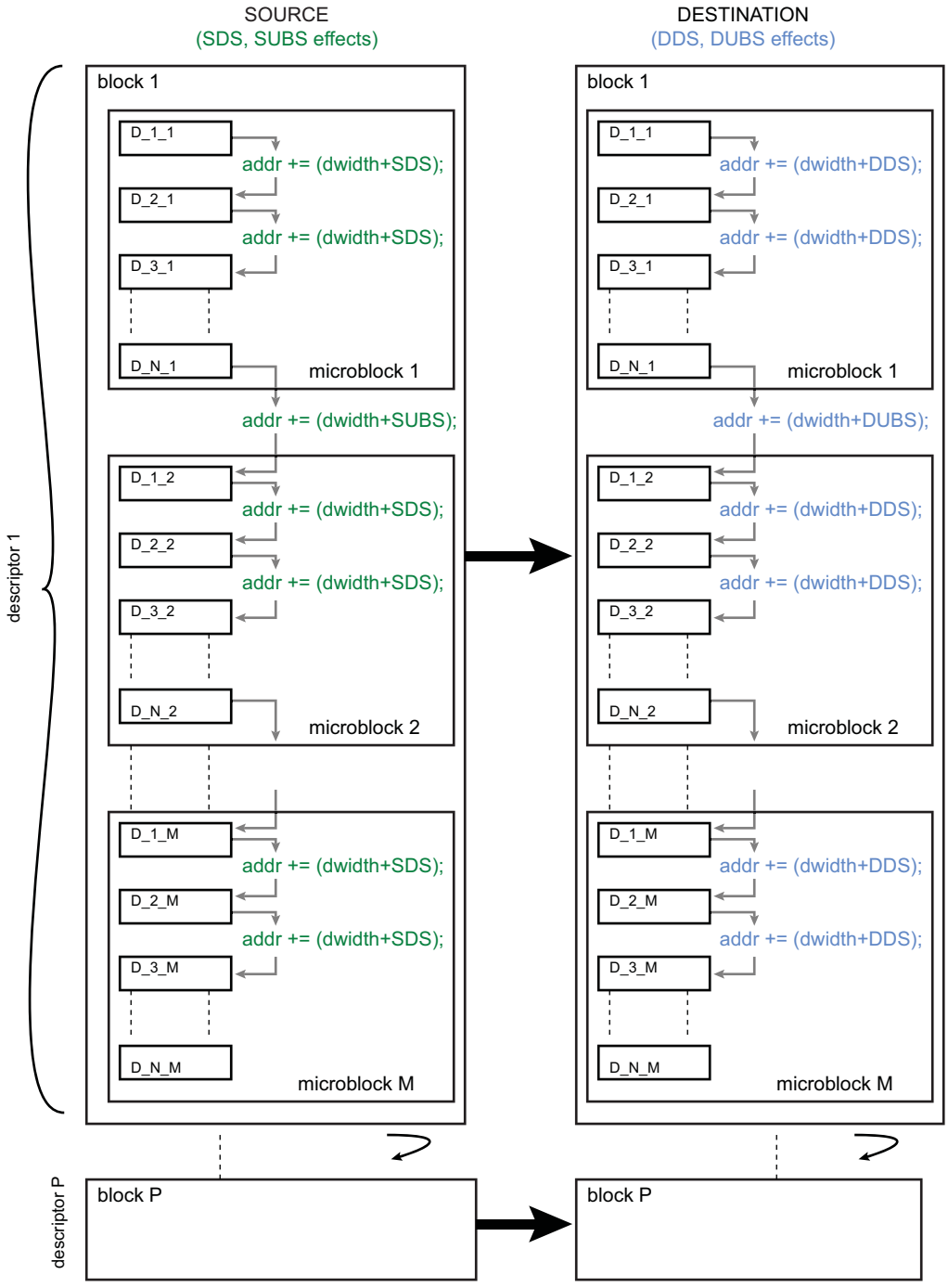
**XDMAC Burst and Incomplete Burst**: In order to improve the overall performance when accessing dynamic external memory, burst access is mandatory. Each data of the microblock is considered as a part of a memory burst. The programmable burst value indicates the largest memory burst allowed on a per channel basis. When the microblock length is not an integral multiple of the burst size, an incomplete burst is performed to read or write the last trailing bytes.

**XDMAC Chunk and Incomplete Chunk**: When a peripheral synchronized transfer is activated, the microblock splits into a number of data chunks. The chunk size is programmable. The larger the chunk is, the better the performance is. When the transfer size is not a multiple of the chunk size, the last chunk may be incomplete.

© 2018 Microchip Technology Inc.

### 37.5.2 Data Striding Diagram

**Figure 37-2. Data Striding Diagram**



### 37.5.3 Transfer Hierarchy Diagrams

**Figure 37-3. XDMAC Memory Transfer Hierarchy**

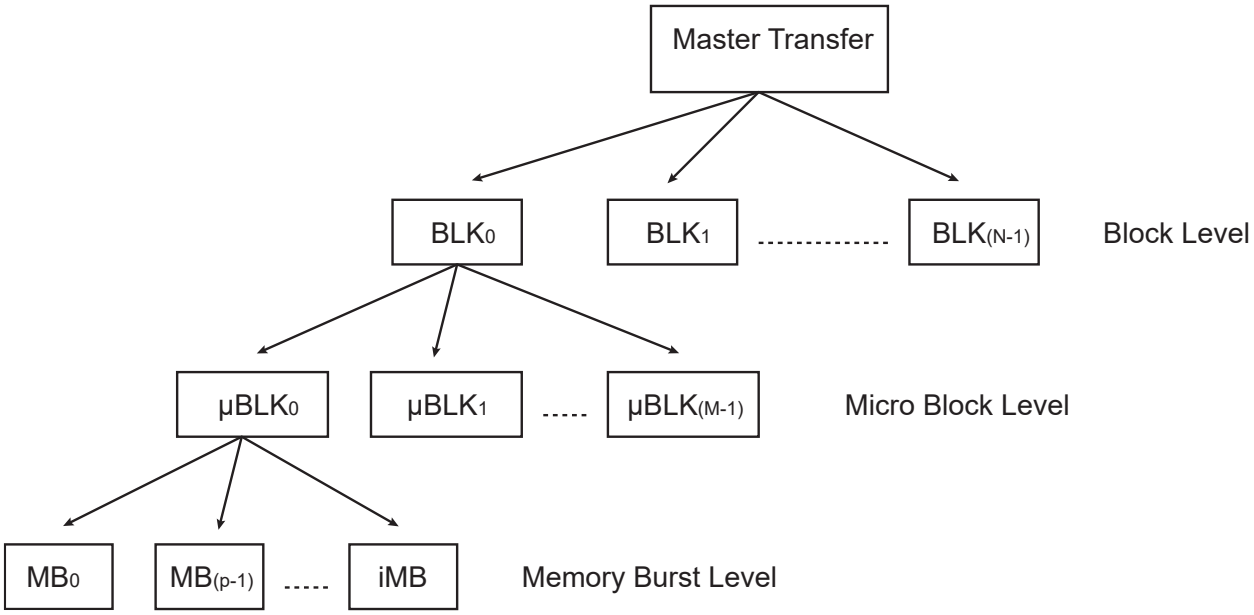
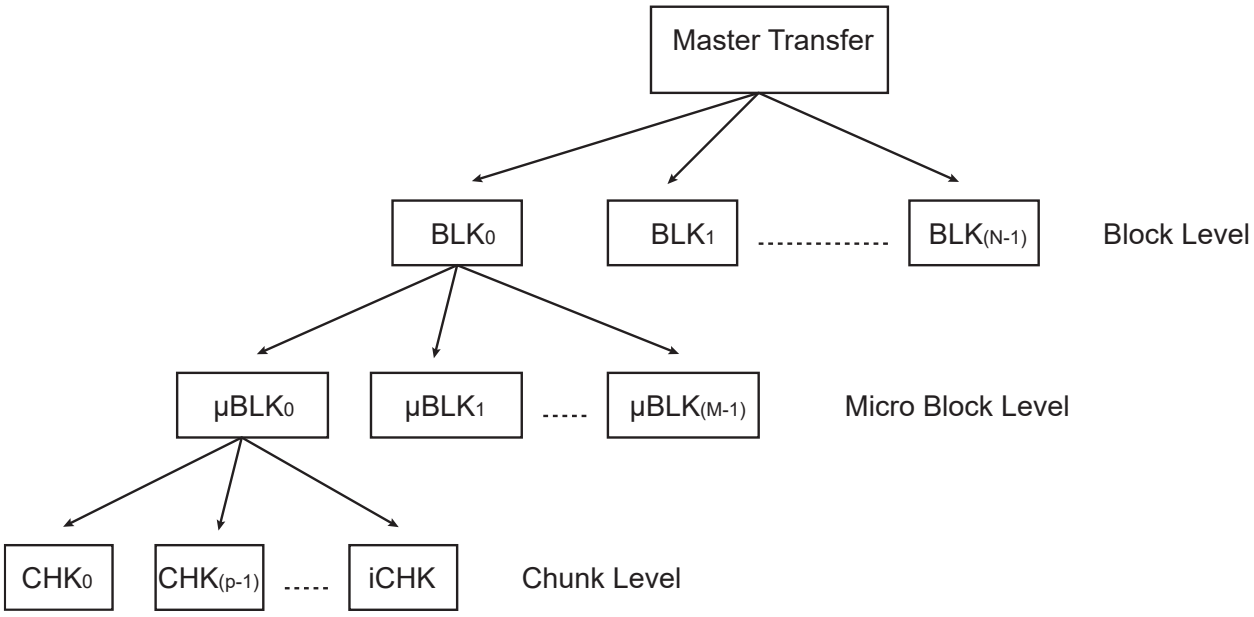



**Figure 37-4. XDAMC Peripheral Transfer Hierarchy**




### 37.5.4 Peripheral Synchronized Transfer

A peripheral hardware request interface is used to control the pace of the chunk transfer. When a peripheral is ready to transmit or receive a chunk of data, it asserts its request line and the DMA Controller transfers a data to or from the memory to the peripheral.

#### 37.5.4.1 Software Triggered Synchronized Transfer

The Peripheral hardware request can be software controlled using the SWREQ field of the XDMAC Global Channel Software Request Register (XDMAC_GSWR). The peripheral synchronized transfer is paced using a processor write access in the XDMAC_GSWR. Each bit of that register triggers a transfer

request. The XDMAC Global Channel Software Request Status Register (XDMAC_GSWS) indicates the status of the request; when set, the request is still pending.

### 37.5.5 XDMAC Transfer Software Operation

**Note:**
When a memory-to-memory transfer is performed, configure the field XDMAC_CCx.PERID (where 'x' is the index of the channel used for transfer) to an unused peripheral ID (refer to table "Peripheral Identifiers").

#### 37.5.5.1 Single Block Transfer With Single Microblock

1. Read the XDMAC Global Channel Status Register (XDMAC_GS) to select a free channel.
2. Clear the pending Interrupt Status bit(s) by reading the selected XDMAC Channel x Interrupt Status Register (XDMAC_CISx).
3. Write the XDMAC Channel x Source Address Register (XDMAC_CSAx) for channel x.
4. Write the XDMAC Channel x Destination Address Register (XDMAC_CDAx) for channel x.
5. Program field UBLEN in the XDMAC Channel x Microblock Control Register (XDMAC_CUBCx) with the number of data.
6. Program the XDMAC Channel x Configuration Register (XDMAC_CCx):
    6.1. Clear XDMAC_CCx.TYPE for a memory-to-memory transfer, otherwise set this bit.
    6.2. Configure XDMAC_CCx.MBSIZE to the memory burst size used.
    6.3. Configure XDMAC_CCx.SAM and DAM to Memory Addressing mode.
    6.4. Configure XDMAC_CCx.DSYNC to select the peripheral transfer direction.
    6.5. Set XDMAC_CCx.PROT to activate a secure channel.
    6.6. Configure XDMAC_CCx.CSIZE to configure the channel chunk size (only relevant for peripheral synchronized transfer).
    6.7. Configure XDMAC_CCx.DWIDTH to configure the transfer data width.
    6.8. Configure XDMAC_CCx.SIF, XDMAC_CCx.DIF to configure the master interface used to read data and write data, respectively.
    6.9. Configure XDMAC_CCx.PERID to select the active hardware request line (only relevant for a peripheral synchronized transfer).
    6.10. Set XDMAC_CCx.SWREQ to use a software request (only relevant for a peripheral synchronized transfer).
7. Clear the following five registers:
    – XDMAC Channel x Next Descriptor Control Register (XDMAC_CNDCx)
    – XDMAC Channel x Block Control Register (XDMAC_CBCx)
    – XDMAC Channel x Data Stride Memory Set Pattern Register (XDMAC_CDS_MSPx)
    – XDMAC Channel x Source Microblock Stride Register (XDMAC_CSUSx)
    – XDMAC Channel x Destination Microblock Stride Register (XDMAC_CDUSx)
    This indicates that the linked list is disabled, there is only one block and striding is disabled.
8. Enable the Microblock interrupt by writing a '1' to bit BIE in the XDMAC Channel x Interrupt Enable Register (XDMAC_CIEx). Enable the Channel x Interrupt Enable bit by writing a '1' to bit IEx in the XDMAC Global Interrupt Enable Register (XDMAC_GIE).
9. Enable channel x by writing a '1' to bit ENx in the XDMAC Global Channel Enable Register (XDMAC_GE). XDMAC_GS.STx (XDMAC Channel x Status bit) is set by hardware.

10. Once completed, the DMA channel sets XDMAC_CISx.BIS (End of Block Interrupt Status bit) and generates an interrupt. XDMAC_GS.STx is cleared by hardware. The software can either wait for an interrupt or poll the channel status bit.

#### 37.5.5.2 Single Block Transfer With Multiple Microblock

1. Read the XDMAC_GS register to choose a free channel.
2. Clear the pending Interrupt Status bit by reading the chosen XDMAC_CISx register.
3. Write the XDMAC_CSAx register for channel x.
4. Write the XDMAC_CDAx register for channel x.
5. Program XDMAC_CUBCx.UBLEN with the number of data.
6. Program XDMAC_CCx register (see "Single Block Transfer With Single Microblock").
7. Program XDMAC_CBCx.BLEN with the number of microblocks of data.
8. Clear the following registers:
   - XDMAC_CNDCx
   - XDMAC_CDS_MSPx
   - XDMAC_CSUSx XDMAC_CDUSx
     This indicates that the linked list is disabled and striding is disabled.
9. Enable the Block interrupt by writing a '1' to XDMAC_CIEx.BIE, enable the Channel x Interrupt Enable bit by writing a '1' to XDMAC_GIEx.IEx.
10. Enable channel x by writing a '1' to the XDMAC_GE.ENx. XDMAC_GS.STx is set by hardware.
11. Once completed, the DMA channel sets XDMAC_CISx.BIS (End of Block Interrupt Status bit) and generates an interrupt. XDMAC_GS.STx is cleared by hardware. The software can either wait for an interrupt or poll the channel status bit.

#### 37.5.5.3 Master Transfer

1. Read the XDMAC_GS register to choose a free channel.
2. Clear the pending Interrupt Status bit by reading the chosen XDMAC_CISx register.
3. Build a linked list of transfer descriptors in memory. The descriptor view is programmable on a per descriptor basis. The linked list items structure must be word aligned. MBR_UBC.NDE must be configured to 0 in the last descriptor to terminate the list.
4. Configure field NDA in the XDMAC Channel x Next Descriptor Address Register (XDMAC_CNDAx) with the first descriptor address and bit XDMAC_CNDAx.NDAIF with the master interface identifier.
5. Configure the XDMAC_CNDCx register:
   5.1. Set XDMAC_CNDCx.NDE to enable the descriptor fetch.
   5.2. Set XDMAC_CNDCx.NDSUP to update the source address at the descriptor fetch time, otherwise clear this bit.
   5.3. Set XDMAC_CNDCx.NDDUP to update the destination address at the descriptor fetch time, otherwise clear this bit.
   5.4. Configure XDMAC_CNDCx.NDVIEW to define the length of the first descriptor.
6. Enable the End of Linked List interrupt by writing a '1' to XDMAC_CIEx.LIE.
7. Enable channel x by writing a '1' to XDMAC_GE.ENx. XDMAC_GS.STx is set by hardware.
8. Once completed, the DMA channel sets XDMAC_CISx.BIS (End of Block Interrupt Status bit) and generates an interrupt. XDMAC_GS.STx is cleared by hardware. The software can either wait for an interrupt or poll the channel status bit.

#### 37.5.5.4 Disabling A Channel Before Transfer Completion

Under normal operation, the software enables a channel by writing a ‘1’ to XDMAC_GE.ENx, then the hardware disables a channel on transfer completion by clearing bit XDMAC_GS.STx. To disable a channel, write a ‘1’ to bit XDMAC_GD.DIx and poll the XDMAC_GS register.

## 37.6 Linked List Descriptor Operation

### 37.6.1 Linked List Descriptor View

#### 37.6.1.1 Channel Next Descriptor View 0–3 Structures

**Table 37-4. Channel Next Descriptor View 0–3 Structures**

| Channel Next Descriptor | Offset | Structure member | Name |
|---|---|---|---|
| View 0 Structure | DSCR_ADDR+0x00 | Next Descriptor Address Member | MBR_NDA |
| | DSCR_ADDR+0x04 | Microblock Control Member | MBR_UBC |
| | DSCR_ADDR+0x08 | Transfer Address Member | MBR_TA |
| View 1 Structure | DSCR_ADDR+0x00 | Next Descriptor Address Member | MBR_NDA |
| | DSCR_ADDR+0x04 | Microblock Control Member | MBR_UBC |
| | DSCR_ADDR+0x08 | Source Address Member | MBR_SA |
| | DSCR_ADDR+0x0C | Destination Address Member | MBR_DA |
| View 2 Structure | DSCR_ADDR+0x00 | Next Descriptor Address Member | MBR_NDA |
| | DSCR_ADDR+0x04 | Microblock Control Member | MBR_UBC |
| | DSCR_ADDR+0x08 | Source Address Member | MBR_SA |
| | DSCR_ADDR+0x0C | Destination Address Member | MBR_DA |
| | DSCR_ADDR+0x10 | Configuration Register | MBR_CFG |
| View 3 Structure | DSCR_ADDR+0x00 | Next Descriptor Address Member | MBR_NDA |
| | DSCR_ADDR+0x04 | Microblock Control Member | MBR_UBC |
| | DSCR_ADDR+0x08 | Source Address Member | MBR_SA |
| | DSCR_ADDR+0x0C | Destination Address Member | MBR_DA |
| | DSCR_ADDR+0x10 | Configuration Member | MBR_CFG |
| | DSCR_ADDR+0x14 | Block Control Member | MBR_BC |
| | DSCR_ADDR+0x18 | Data Stride Member | MBR_DS |
| | DSCR_ADDR+0x1C | Source Microblock Stride Member | MBR_SUS |
| | DSCR_ADDR+0x20 | Destination Microblock Stride Member | MBR_DUS |

### 37.6.2 Descriptor Structure Members Description

#### 37.6.2.1 Descriptor Structure Microblock Control Member

**Name:** MBR_UBC
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | NVIEW[1:0] | | NDEN | NSEN | NDE |
| Access | | | | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UBLEN[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UBLEN[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UBLEN[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

**Bits 28:27 – NVIEW[1:0]** Next Descriptor View

| Value | Name | Description |
|---|---|---|
| 0 | NDV0 | Next Descriptor View 0 |
| 1 | NDV1 | Next Descriptor View 1 |
| 2 | NDV2 | Next Descriptor View 2 |
| 3 | NDV3 | Next Descriptor View 3 |

**Bit 26 – NDEN** Next Descriptor Destination Update

| Value | Description |
|---|---|
| 0 | Destination parameters remain unchanged. |
| 1 | Destination parameters are updated when the descriptor is retrieved. |

**Bit 25 – NSEN** Next Descriptor Source Update

| Value | Description |
|---|---|
| 0 | Source parameters remain unchanged. |
| 1 | Source parameters are updated when the descriptor is retrieved. |

**Bit 24 – NDE** Next Descriptor Enable

| Value | Description |
|---|---|
| 0 | Descriptor fetch is disabled. |
| 1 | Descriptor fetch is enabled. |

**Bits 23:0 – UBLEN[23:0]** Microblock Length
This field indicates the number of data in the microblock. The microblock contains UBLEN data.

## 37.7 XDMAC Maintenance Software Operations

### 37.7.1 Disabling a Channel

A disable channel request occurs when a write operation is performed in the XDMAC_GD register. If the channel is source peripheral synchronized (bit XDMAC_CCx.TYPE is set and bit XDMAC_CCx.DSYNC is cleared), then pending bytes (bytes located in the FIFO) are written to memory and bit XDMAC_CISx.DIS is set. If the channel is not source peripheral synchronized, the current channel transaction (read or write) is terminated and XDMAC_CISx.DIS is set. XDMAC_GS.STx is cleared by hardware when the current transfer is completed. The channel is no longer active and can be reused.

### 37.7.2 Suspending a Channel

A disable channel request occurs when a write operation is performed in the XDMAC_GD register. If the channel is source peripheral synchronized (bit XDMAC_CCx.TYPE is set and bit XDMAC_CCx.DSYNC is cleared), then pending bytes (bytes located in the FIFO) are written to memory and bit XDMAC_CISx.DIS is set. If the channel is not source peripheral synchronized, the current channel transaction (read or write) is terminated and XDMAC_CISx.DIS is set. XDMAC_GS.STx is cleared by hardware when the current transfer is completed. The channel is no longer active and can be reused.

### 37.7.3 Flushing a Channel

A FIFO flush command is issued by writing to the XDMAC_SWF register. The content of the FIFO is written to memory. XDMAC_CISx.FIS (End of Flush Interrupt Status bit) is set when the last byte is successfully transferred to memory. The channel is not disabled. The flush operation is not blocking, meaning that read operation can be scheduled during the flush write operation. The flush operation is only relevant for peripheral to memory transfer where pending peripheral bytes are buffered into the channel FIFO.

### 37.7.4 Maintenance Operation Priority

#### 37.7.4.1 Disable Operation Priority

- When a disable request occurs on a suspended channel, the XDMAC_GWS.WSx (Channel x Write Suspend bit) is cleared. If the transfer is source peripheral synchronized, the pending bytes are drained to memory. The bit XDMAC_CISx.DIS is set.
- When a disable request follows a flush request, if the flush last transaction is not yet scheduled, the flush request is discarded and the disable procedure is applied. Bit XDMAC_CISx.FIS is not set. Bit XDMAC_CISx.DIS is set when the disable request is completed. If the flush request transaction is already scheduled, the XDMAC_CISx.FIS is set. XDMAC_CISx.DIS is also set when the disable request is completed.

#### 37.7.4.2 Flush Operation Priority

- When a flush request occurs on a suspended channel, if there are pending bytes in the FIFO, they are written out to memory, XDMAC_CISx.FIS is set. If the FIFO is empty, XDMAC_CISx.FIS is also set.
- If the flush operation is performed after a disable request, the flush command is ignored. XDMAC_CISx.FIS is not set.

#### 37.7.4.3 Suspend Operation Priority

If the suspend operation is performed after a disable request, the write suspend operation is ignored.

## 37.8 XDMAC Software Requirements

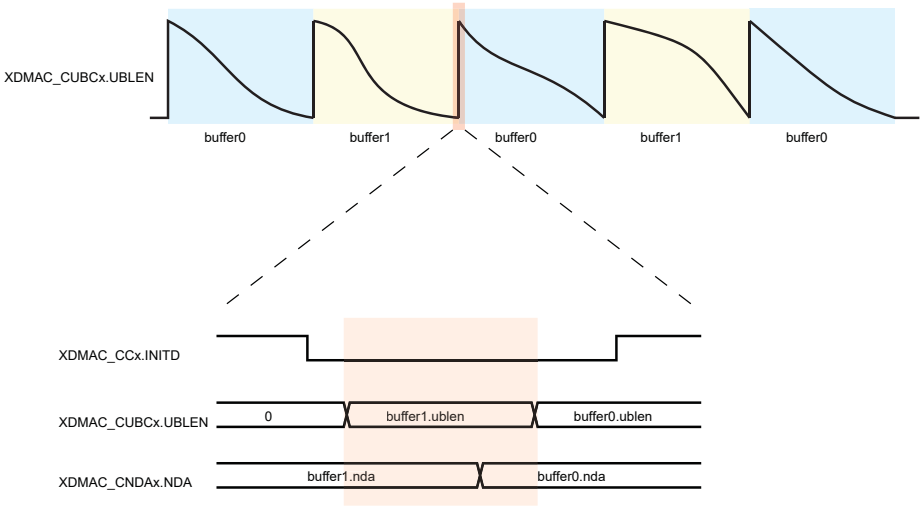
- Write operations to channel registers are not be performed in an active channel after the channel is enabled. If any channel parameters must be reprogrammed, this can only be done after disabling the XDMAC channel.
- XDMAC_CSAx and XDMAC_CDAx channel registers must be programmed with a byte, half-word, word or double-word aligned address depending on the Channel x Data Width field (DWIDTH) of the XDMAC Channel x Configuration Register. When a memory-to-peripheral transfer is performed, the XDMAC_CSAx address register has no alignment requirement.
- When a memory-to-memory transfer is performed, configure the field XDMAC_CCx.PERID (where ‘x’ is the index of the channel used for the transfer) to an unused peripheral ID (refer to table “Peripheral Identifiers”).
- When XDMAC_CC.INITD is set to 0, XDMAC_CUBC.UBLEN and XDMAC_CNDA.NDA field values are unreliable when the descriptor is being updated. The following procedure applies to get the buffer descriptor identifier and the residual bytes:

```
Read XDMAC_CNDAx.NDA(nda0)
Read XDMAC_CCx.INITD(initd0)
Read XDMAC_CCx.INITD(initd0)
Read XDMAC_CUBCx.UBLEN(ublen)
Read XDMAC_CCx.INITD(initd1)
Read XDMA_CNDAx.NDA(nda1)
If (nda0 == nda1 && initd0 == 1 && initd1 == 1).
Then the ublen is correct, the buffer id is nda.
Else retry
```

  See the figure below.

- Each DMA channel can be configured in either Secure or Non-secure mode independently. When a DMA channel is secure, its configuration registers cannot be modified by non-secure software. The configuration registers of a secure channel are read as zero by non-secure software and the read-only registers are also read as zero.

**Figure 37-5. INITD Timing Diagram**



## 37.9 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | XDMAC_GTYPE | 7:0 | FIFO_SZ[2:0] | | | NB_CH[4:0] | | | | |
| | | 15:8 | FIFO_SZ[10:3] | | | | | | | |
| | | 23:16 | | NB_REQ[6:0] | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | XDMAC_GCFG | 7:0 | | | | | CGDISIF | CGDISFIFO | CGDISPIPE | CGDISREG |
| | | 15:8 | | | | | | | | BXKBEN |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | XDMAC_GWAC | 7:0 | PW1[3:0] | | | | PW0[3:0] | | | |
| | | 15:8 | PW3[3:0] | | | | PW2[3:0] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | XDMAC_GIE | 7:0 | IE7 | IE6 | IE5 | IE4 | IE3 | IE2 | IE1 | IE0 |
| | | 15:8 | IE15 | IE14 | IE13 | IE12 | IE11 | IE10 | IE9 | IE8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | XDMAC_GID | 7:0 | ID7 | ID6 | ID5 | ID4 | ID3 | ID2 | ID1 | ID0 |
| | | 15:8 | ID15 | ID14 | ID13 | ID12 | ID11 | ID10 | ID9 | ID8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | XDMAC_GIM | 7:0 | IM7 | IM6 | IM5 | IM4 | IM3 | IM2 | IM1 | IM0 |
| | | 15:8 | IM15 | IM14 | IM13 | IM12 | IM11 | IM10 | IM9 | IM8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | XDMAC_GIS | 7:0 | IS7 | IS6 | IS5 | IS4 | IS3 | IS2 | IS1 | IS0 |
| | | 15:8 | IS15 | IS14 | IS13 | IS12 | IS11 | IS10 | IS9 | IS8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | XDMAC_GE | 7:0 | EN7 | EN6 | EN5 | EN4 | EN3 | EN2 | EN1 | EN0 |
| | | 15:8 | EN15 | EN14 | EN13 | EN12 | EN11 | EN10 | EN9 | EN8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | XDMAC_GD | 7:0 | DI7 | DI6 | DI5 | DI4 | DI3 | DI2 | DI1 | DI0 |
| | | 15:8 | DI15 | DI14 | DI13 | DI12 | DI11 | DI10 | DI9 | DI8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 | XDMAC_GS | 7:0 | ST7 | ST6 | ST5 | ST4 | ST3 | ST2 | ST1 | ST0 |
| | | 15:8 | ST15 | ST14 | ST13 | ST12 | ST11 | ST10 | ST9 | ST8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x28 | XDMAC_GRS | 7:0 | RS7 | RS6 | RS5 | RS4 | RS3 | RS2 | RS1 | RS0 |
| | | 15:8 | RS15 | RS14 | RS13 | RS12 | RS11 | RS10 | RS9 | RS8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x2C | XDMAC_GWS | 7:0 | WS7 | WS6 | WS5 | WS4 | WS3 | WS2 | WS1 | WS0 |
| | | 15:8 | WS15 | WS14 | WS13 | WS12 | WS11 | WS10 | WS9 | WS8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x30 | XDMAC_GRWS | 7:0 | RWS7 | RWS6 | RWS5 | RWS4 | RWS3 | RWS2 | RWS1 | RWS0 |
| | | 15:8 | RWS15 | RWS14 | RWS13 | RWS12 | RWS11 | RWS10 | RWS9 | RWS8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x34 | XDMAC_GRWR | 7:0 | RWR7 | RWR6 | RWR5 | RWR4 | RWR3 | RWR2 | RWR1 | RWR0 |
| | | 15:8 | RWR15 | RWR14 | RWR13 | RWR12 | RWR11 | RWR10 | RWR9 | RWR8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x38 | XDMAC_GSWR | 7:0 | SWREQ7 | SWREQ6 | SWREQ5 | SWREQ4 | SWREQ3 | SWREQ2 | SWREQ1 | SWREQ0 |
| | | 15:8 | SWREQ15 | SWREQ14 | SWREQ13 | SWREQ12 | SWREQ11 | SWREQ10 | SWREQ9 | SWREQ8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x3C | XDMAC_GSWS | 7:0 | SWRS7 | SWRS6 | SWRS5 | SWRS4 | SWRS3 | SWRS2 | SWRS1 | SWRS0 |
| | | 15:8 | SWRS15 | SWRS14 | SWRS13 | SWRS12 | SWRS11 | SWRS10 | SWRS9 | SWRS8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x40 | XDMAC_GSWF | 7:0 | SWF7 | SWF6 | SWF5 | SWF4 | SWF3 | SWF2 | SWF1 | SWF0 |
| | | 15:8 | SWF15 | SWF14 | SWF13 | SWF12 | SWF11 | SWF10 | SWF9 | SWF8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x44 ... 0x4F | Reserved | | | | | | | | | |
| 0x50 | XDMAC_CIE0 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x54 | XDMAC_CID0 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x58 | XDMAC_CIM0 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x5C | XDMAC_CIS0 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x60 | XDMAC_CSA0 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x64 | XDMAC_CDA0 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x68 | XDMAC_CNDA0 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x6C | XDMAC_CNDC0 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x70 | XDMAC_CUBC0 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x74 | XDMAC_CBC0 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x78 | XDMAC_CC0 | 7:0 | MEMSET | SWREQ | PROT | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |
| 0x7C | XDMAC_CDS_MSP0 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x80 | XDMAC_CSUS0 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x84 | XDMAC_CDUS0 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x88 ... 0x8F | Reserved | | | | | | | | | |
| 0x90 | XDMAC_CIE1 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x94 | XDMAC_CID1 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x98 | XDMAC_CIM1 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x9C | XDMAC_CIS1 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xA0 | XDMAC_CSA1 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0xA4 | XDMAC_CDA1 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0xA8 | XDMAC_CNDA1 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0xAC | XDMAC_CNDC1 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xB0 | XDMAC_CUBC1 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xB4 | XDMAC_CBC1 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xB8 | XDMAC_CC1 | 7:0 | MEMSET | SWREQ | PROT | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |
| 0xBC | XDMAC_CDS_MSP1 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0xC0 | XDMAC_CSUS1 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xC4 | XDMAC_CDUS1 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xC8 ... 0xCF | Reserved | | | | | | | | | |
| 0xD0 | XDMAC_CIE2 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xD4 | XDMAC_CID2 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xD8 | XDMAC_CIM2 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xDC | XDMAC_CIS2 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xE0 | XDMAC_CSA2 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0xE4 | XDMAC_CDA2 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xE8 | XDMAC_CNDA2 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0xEC | XDMAC_CNDC2 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xF0 | XDMAC_CUBC2 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xF4 | XDMAC_CBC2 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xF8 | XDMAC_CC2 | 7:0 | MEMSET | SWREQ | PROT | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |
| 0xFC | XDMAC_CDS_MSP2 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0100 | XDMAC_CSUS2 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0104 | XDMAC_CDUS2 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0108 ... 0x010F | Reserved | | | | | | | | | |
| 0x0110 | XDMAC_CIE3 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0114 | XDMAC_CID3 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0118 | XDMAC_CIM3 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x011C | XDMAC_CIS3 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0120 | XDMAC_CSA3 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0124 | XDMAC_CDA3 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0128 | XDMAC_CNDA3 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x012C | XDMAC_CNDC3 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0130 | XDMAC_CUBC3 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0134 | XDMAC_CBC3 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0138 | XDMAC_CC3 | 7:0 | MEMSET | SWREQ | PROT | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |
| 0x013C | XDMAC_CDS_MSP3 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0140 | XDMAC_CSUS3 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0144 | XDMAC_CDUS3 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0148 ... 0x014F | Reserved | | | | | | | | | |
| 0x0150 | XDMAC_CIE4 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0154 | XDMAC_CID4 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0158 | XDMAC_CIM4 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x015C | XDMAC_CIS4 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0160 | XDMAC_CSA4 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0164 | XDMAC_CDA4 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0168 | XDMAC_CNDA4 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x016C | XDMAC_CNDC4 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0170 | XDMAC_CUBC4 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0174 | XDMAC_CBC4 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0178 | XDMAC_CC4 | 7:0 | MEMSET | SWREQ | PROT | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |
| 0x017C | XDMAC_CDS_MSP4 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0180 | XDMAC_CSUS4 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0184 | XDMAC_CDUS4 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0188 ... 0x018F | Reserved | | | | | | | | | |
| 0x0190 | XDMAC_CIE5 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0194 | XDMAC_CID5 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0198 | XDMAC_CIM5 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x019C | XDMAC_CIS5 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01A0 | XDMAC_CSA5 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x01A4 | XDMAC_CDA5 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x01A8 | XDMAC_CNDA5 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x01AC | XDMAC_CNDC5 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01B0 | XDMAC_CUBC5 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01B4 | XDMAC_CBC5 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01B8 | XDMAC_CC5 | 7:0 | MEMSET | SWREQ | PROT | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |
| 0x01BC | XDMAC_CDS_MSP5 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x01C0 | XDMAC_CSUS5 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01C4 | XDMAC_CDUS5 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01C8 ... 0x01CF | Reserved | | | | | | | | | |
| 0x01D0 | XDMAC_CIE6 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x01D4 | XDMAC_CID6 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01D8 | XDMAC_CIM6 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01DC | XDMAC_CIS6 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01E0 | XDMAC_CSA6 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x01E4 | XDMAC_CDA6 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x01E8 | XDMAC_CNDA6 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x01EC | XDMAC_CNDC6 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01F0 | XDMAC_CUBC6 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01F4 | XDMAC_CBC6 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01F8 | XDMAC_CC6 | 7:0 | MEMSET | SWREQ | PROT | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |
| 0x01FC | XDMAC_CDS_MSP6 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0200 | XDMAC_CSUS6 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0204 | XDMAC_CDUS6 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0208 ... 0x020F | Reserved | | | | | | | | | |
| 0x0210 | XDMAC_CIE7 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0214 | XDMAC_CID7 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0218 | XDMAC_CIM7 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x021C | XDMAC_CIS7 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0220 | XDMAC_CSA7 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0224 | XDMAC_CDA7 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0228 | XDMAC_CNDA7 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x022C | XDMAC_CNDC7 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0230 | XDMAC_CUBC7 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0234 | XDMAC_CBC7 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0238 | XDMAC_CC7 | 7:0 | MEMSET | SWREQ | PROT | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |
| 0x023C | XDMAC_CDS_MSP7 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0240 | XDMAC_CSUS7 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0244 | XDMAC_CDUS7 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0248 ... 0x024F | Reserved | | | | | | | | | |
| 0x0250 | XDMAC_CIE8 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0254 | XDMAC_CID8 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0258 | XDMAC_CIM8 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x025C | XDMAC_CIS8 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0260 | XDMAC_CSA8 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0264 | XDMAC_CDA8 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0268 | XDMAC_CNDA8 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x026C | XDMAC_CNDC8 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0270 | XDMAC_CUBC8 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0274 | XDMAC_CBC8 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0278 | XDMAC_CC8 | 7:0 | MEMSET | SWREQ | PROT | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |
| 0x027C | XDMAC_CDS_MSP8 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0280 | XDMAC_CSUS8 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0284 | XDMAC_CDUS8 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0288 ... 0x028F | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0290 | XDMAC_CIE9 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0294 | XDMAC_CID9 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0298 | XDMAC_CIM9 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x029C | XDMAC_CIS9 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02A0 | XDMAC_CSA9 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x02A4 | XDMAC_CDA9 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x02A8 | XDMAC_CNDA9 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x02AC | XDMAC_CNDC9 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02B0 | XDMAC_CUBC9 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02B4 | XDMAC_CBC9 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02B8 | XDMAC_CC9 | 7:0 | MEMSET | SWREQ | PROT | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x02BC | XDMAC_CDS_MSP9 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x02C0 | XDMAC_CSUS9 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02C4 | XDMAC_CDUS9 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02C8 ... 0x02CF | Reserved | | | | | | | | | |
| 0x02D0 | XDMAC_CIE10 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02D4 | XDMAC_CID10 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02D8 | XDMAC_CIM10 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02DC | XDMAC_CIS10 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02E0 | XDMAC_CSA10 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x02E4 | XDMAC_CDA10 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x02E8 | XDMAC_CNDA10 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x02EC | XDMAC_CNDC10 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02F0 | XDMAC_CUBC10 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02F4 | XDMAC_CBC10 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02F8 | XDMAC_CC10 | 7:0 | MEMSET | SWREQ | PROT | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |
| 0x02FC | XDMAC_CDS_MSP10 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0300 | XDMAC_CSUS10 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0304 | XDMAC_CDUS10 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0308 ... 0x030F | Reserved | | | | | | | | | |
| 0x0310 | XDMAC_CIE11 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0314 | XDMAC_CID11 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0318 | XDMAC_CIM11 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x031C | XDMAC_CIS11 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0320 | XDMAC_CSA11 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0324 | XDMAC_CDA11 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0328 | XDMAC_CNDA11 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x032C | XDMAC_CNDC11 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0330 | XDMAC_CUBC11 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0334 | XDMAC_CBC11 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0338 | XDMAC_CC11 | 7:0 | MEMSET | SWREQ | PROT | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |
| 0x033C | XDMAC_CDS_MSP11 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0340 | XDMAC_CSUS11 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0344 | XDMAC_CDUS11 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0348 ... 0x034F | Reserved | | | | | | | | | |
| 0x0350 | XDMAC_CIE12 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0354 | XDMAC_CID12 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0358 | XDMAC_CIM12 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x035C | XDMAC_CIS12 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0360 | XDMAC_CSA12 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0364 | XDMAC_CDA12 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0368 | XDMAC_CNDA12 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x036C | XDMAC_CNDC12 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0370 | XDMAC_CUBC12 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0374 | XDMAC_CBC12 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0378 | XDMAC_CC12 | 7:0 | MEMSET | SWREQ | PROT | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |
| 0x037C | XDMAC_CDS_MSP12 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0380 | XDMAC_CSUS12 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0384 | XDMAC_CDUS12 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0388 ... 0x038F | Reserved | | | | | | | | | |
| 0x0390 | XDMAC_CIE13 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0394 | XDMAC_CID13 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0398 | XDMAC_CIM13 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x039C | XDMAC_CIS13 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03A0 | XDMAC_CSA13 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x03A4 | XDMAC_CDA13 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x03A8 | XDMAC_CNDA13 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x03AC | XDMAC_CNDC13 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03B0 | XDMAC_CUBC13 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03B4 | XDMAC_CBC13 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03B8 | XDMAC_CC13 | 7:0 | MEMSET | SWREQ | PROT | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |
| 0x03BC | XDMAC_CDS_MSP13 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x03C0 | XDMAC_CSUS13 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03C4 | XDMAC_CDUS13 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03C8 ... 0x03CF | Reserved | | | | | | | | | |
| 0x03D0 | XDMAC_CIE14 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03D4 | XDMAC_CID14 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x03D8 | XDMAC_CIM14 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03DC | XDMAC_CIS14 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03E0 | XDMAC_CSA14 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x03E4 | XDMAC_CDA14 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x03E8 | XDMAC_CNDA14 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x03EC | XDMAC_CNDC14 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03F0 | XDMAC_CUBC14 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03F4 | XDMAC_CBC14 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03F8 | XDMAC_CC14 | 7:0 | MEMSET | SWREQ | PROT | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |
| 0x03FC | XDMAC_CDS_MSP14 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0400 | XDMAC_CSUS14 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0404 | XDMAC_CDUS14 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0408 ... 0x040F | Reserved | | | | | | | | | |
| 0x0410 | XDMAC_CIE15 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0414 | XDMAC_CID15 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0418 | XDMAC_CIM15 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x041C | XDMAC_CIS15 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0420 | XDMAC_CSA15 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0424 | XDMAC_CDA15 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0428 | XDMAC_CNDA15 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x042C | XDMAC_CNDC15 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0430 | XDMAC_CUBC15 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0434 | XDMAC_CBC15 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0438 | XDMAC_CC15 | 7:0 | MEMSET | SWREQ | PROT | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |
| 0x043C | XDMAC_CDS_MSP15 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0440 | XDMAC_CSUS15 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0444 | XDMAC_CDUS15 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |

### 37.9.1 XDMAC Global Type Register

**Name:** XDMAC_GTYPE
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | NB_REQ[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FIFO_SZ[10:3] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FIFO_SZ[2:0] | | | NB_CH[4:0] | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 22:16 – NB_REQ[6:0]** Number of Peripheral Requests Minus One

**Bits 15:5 – FIFO_SZ[10:0]** Number of Bytes

**Bits 4:0 – NB_CH[4:0]** Number of Channels Minus One

### 37.9.2 XDMAC Global Configuration Register

**Name:** XDMAC_GCFG
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | BXKBEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CGDISIF | CGDISFIFO | CGDISPIPE | CGDISREG |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 8 – BXKBEN** Boundary X Kilobyte Enable

| Value | Description |
|---|---|
| 0 | The 1 Kbyte boundary is used. |
| 1 | The controller does not meet the AHB specification. |

**Bit 3 – CGDISIF** Bus Interface Clock Gating Disable

| Value | Description |
|---|---|
| 0 | The automatic clock gating is enabled for the system bus interface. |
| 1 | The automatic clock gating is disabled for the system bus interface. |

**Bit 2 – CGDISFIFO** FIFO Clock Gating Disable

| Value | Description |
|---|---|
| 0 | The automatic clock gating is enabled for the main FIFO. |
| 1 | The automatic clock gating is disabled for the main FIFO. |

**Bit 1 – CGDISPIPE** Pipeline Clock Gating Disable

| Value | Description |
|---|---|
| 0 | The automatic clock gating is enabled for the main pipeline. |
| 1 | The automatic clock gating is disabled for the main pipeline. |

**Bit 0 – CGDISREG** Configuration Registers Clock Gating Disable

| Value | Description |
|---|---|
| 0 | The automatic clock gating is enabled for the configuration registers. |
| 1 | The automatic clock gating is disabled for the configuration registers. |

### 37.9.3 XDMAC Global Weighted Arbiter Configuration Register

**Name:** XDMAC_GWAC
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PW3[3:0] | | | | PW2[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PW1[3:0] | | | | PW0[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:12 – PW3[3:0]** Pool Weight 3
This field indicates the weight of pool 3 in the arbitration scheme of the DMA scheduler.

**Bits 11:8 – PW2[3:0]** Pool Weight 2
This field indicates the weight of pool 2 in the arbitration scheme of the DMA scheduler.

**Bits 7:4 – PW1[3:0]** Pool Weight 1
This field indicates the weight of pool 1 in the arbitration scheme of the DMA scheduler.

**Bits 3:0 – PW0[3:0]** Pool Weight 0
This field indicates the weight of pool 0 in the arbitration scheme of the DMA scheduler.

### 37.9.4 XDMAC Global Interrupt Enable Register

**Name:** XDMAC_GIE
**Offset:** 0x0C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IE15 | IE14 | IE13 | IE12 | IE11 | IE10 | IE9 | IE8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IE7 | IE6 | IE5 | IE4 | IE3 | IE2 | IE1 | IE0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – IE** XDMAC Channel x Interrupt Enable

| Value | Description |
|---|---|
| 0 | This bit has no effect. The Channel x Interrupt Mask bit (XDMAC_GIM.IMx) is not modified. |
| 1 | The corresponding mask bit is set. The XDMAC Channel x Interrupt Status register (XDMAC_GIS) can generate an interrupt. |

### 37.9.5 XDMAC Global Interrupt Disable Register

**Name:** XDMAC_GID
**Offset:** 0x10
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ID15 | ID14 | ID13 | ID12 | ID11 | ID10 | ID9 | ID8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ID7 | ID6 | ID5 | ID4 | ID3 | ID2 | ID1 | ID0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – ID** XDMAC Channel x Interrupt Disable

| Value | Description |
|---|---|
| 0 | This bit has no effect. The Channel x Interrupt Mask bit (XDMAC_GIM.IMx) is not modified. |
| 1 | The corresponding mask bit is reset. The Channel x Interrupt Status register interrupt (XDMAC_GIS) is masked. |

### 37.9.6 XDMAC Global Interrupt Mask Register

**Name:** XDMAC_GIM
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IM15 | IM14 | IM13 | IM12 | IM11 | IM10 | IM9 | IM8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IM7 | IM6 | IM5 | IM4 | IM3 | IM2 | IM1 | IM0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – IM** XDMAC Channel x Interrupt Mask

| Value | Description |
|---|---|
| 0 | This bit indicates that the channel x interrupt source is masked. The interrupt line is not raised. |
| 1 | This bit indicates that the channel x interrupt source is unmasked. |

### 37.9.7 XDMAC Global Interrupt Status Register

**Name:** XDMAC_GIS
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IS15 | IS14 | IS13 | IS12 | IS11 | IS10 | IS9 | IS8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IS7 | IS6 | IS5 | IS4 | IS3 | IS2 | IS1 | IS0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – IS** XDMAC Channel x Interrupt Status

| Value | Description |
|---|---|
| 0 | This bit indicates that either the interrupt source is masked at the channel level or no interrupt is pending for channel x. |
| 1 | This bit indicates that an interrupt is pending for the channel x. |

### 37.9.8 XDMAC Global Channel Enable Register

**Name:** XDMAC_GE
**Offset:** 0x1C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EN15 | EN14 | EN13 | EN12 | EN11 | EN10 | EN9 | EN8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN7 | EN6 | EN5 | EN4 | EN3 | EN2 | EN1 | EN0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – EN** XDMAC Channel x Enable

| Value | Description |
|---|---|
| 0 | This bit has no effect. |
| 1 | Enables channel n. This operation is permitted if the Channel x Status bit (XDMAC_GS.STx) was read as '0'. |

### 37.9.9 XDMAC Global Channel Disable Register

**Name:** XDMAC_GD
**Offset:** 0x20
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DI15 | DI14 | DI13 | DI12 | DI11 | DI10 | DI9 | DI8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DI7 | DI6 | DI5 | DI4 | DI3 | DI2 | DI1 | DI0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – DI** XDMAC Channel x Disable

| Value | Description |
|---|---|
| 0 | This bit has no effect. |
| 1 | Disables channel x. |

### 37.9.10 XDMAC Global Channel Status Register

**Name:** XDMAC_GS
**Offset:** 0x24
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ST15 | ST14 | ST13 | ST12 | ST11 | ST10 | ST9 | ST8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ST7 | ST6 | ST5 | ST4 | ST3 | ST2 | ST1 | ST0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – ST** XDMAC Channel x Status

| Value | Description |
|---|---|
| 0 | This bit indicates that the channel x is disabled. |
| 1 | This bit indicates that the channel x is enabled. If a channel disable request is issued, this bit remains asserted until pending transaction is completed. |

### 37.9.11 XDMAC Global Channel Read Suspend Register

**Name:** XDMAC_GRS
**Offset:** 0x28
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RS15 | RS14 | RS13 | RS12 | RS11 | RS10 | RS9 | RS8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RS7 | RS6 | RS5 | RS4 | RS3 | RS2 | RS1 | RS0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – RSx** XDMAC Channel x Read Suspend

| Value | Description |
|---|---|
| 0 | The read channel is not suspended. |
| 1 | The source requests for channel n are no longer serviced by the system scheduler. |

### 37.9.12 XDMAC Global Channel Write Suspend Register

**Name:** XDMAC_GWS
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WS15 | WS14 | WS13 | WS12 | WS11 | WS10 | WS9 | WS8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WS7 | WS6 | WS5 | WS4 | WS3 | WS2 | WS1 | WS0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – WSx** XDMAC Channel x Write Suspend

| Value | Description |
|---|---|
| 0 | The write channel is not suspended. |
| 1 | Destination requests are no longer routed to the scheduler. |

### 37.9.13 XDMAC Global Channel Read Write Suspend Register

**Name:** XDMAC_GRWS
**Offset:** 0x30
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RWS15 | RWS14 | RWS13 | RWS12 | RWS11 | RWS10 | RWS9 | RWS8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RWS7 | RWS6 | RWS5 | RWS4 | RWS3 | RWS2 | RWS1 | RWS0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – RWSx** XDMAC Channel x Read Write Suspend

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Read and write requests are suspended. |

### 37.9.14 XDMAC Global Channel Read Write Resume Register

**Name:** XDMAC_GRWR
**Offset:** 0x34
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RWR15 | RWR14 | RWR13 | RWR12 | RWR11 | RWR10 | RWR9 | RWR8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RWR7 | RWR6 | RWR5 | RWR4 | RWR3 | RWR2 | RWR1 | RWR0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – RWRx** XDMAC Channel x Read Write Resume

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Read and write requests are serviced. |

### 37.9.15 XDMAC Global Channel Software Request Register

**Name:** XDMAC_GSWR
**Offset:** 0x38
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SWREQ15 | SWREQ14 | SWREQ13 | SWREQ12 | SWREQ11 | SWREQ10 | SWREQ9 | SWREQ8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWREQ7 | SWREQ6 | SWREQ5 | SWREQ4 | SWREQ3 | SWREQ2 | SWREQ1 | SWREQ0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – SWREQ** XDMAC Channel x Software Request

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Requests a DMA transfer for channel x. |

### 37.9.16 XDMAC Global Channel Software Request Status Register

**Name:** XDMAC_GSWS
**Offset:** 0x3C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SWRS15 | SWRS14 | SWRS13 | SWRS12 | SWRS11 | SWRS10 | SWRS9 | SWRS8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWRS7 | SWRS6 | SWRS5 | SWRS4 | SWRS3 | SWRS2 | SWRS1 | SWRS0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – SWRS** XDMAC Channel x Software Request Status

| Value | Description |
|---|---|
| 0 | Channel x source request is serviced. |
| 1 | Channel x source request is pending. |

### 37.9.17 XDMAC Global Channel Software Flush Request Register

**Name:** XDMAC_GSWF
**Offset:** 0x40
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SWF15 | SWF14 | SWF13 | SWF12 | SWF11 | SWF10 | SWF9 | SWF8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWF7 | SWF6 | SWF5 | SWF4 | SWF3 | SWF2 | SWF1 | SWF0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – SWFx** XDMAC Channel x Software Flush Request

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Requests a DMA transfer flush for channel x. This bit is only relevant when the transfer is source peripheral synchronized. |

### 37.9.18 XDMAC Channel x Interrupt Enable Register [x=0..15]

**Name:** XDMAC_CIE
**Offset:** 0x50 + n*0x40 [n=0..15]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| Access | | W | W | W | W | W | W | W |
| Reset | | – | – | – | – | – | – | – |

**Bit 6 – ROIE** Request Overflow Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables request overflow error interrupt. |

**Bit 5 – WBIE** Write Bus Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables write bus error interrupt. |

**Bit 4 – RBIE** Read Bus Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables read bus error interrupt. |

**Bit 3 – FIE** End of Flush Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables end of flush interrupt. |

**Bit 2 – DIE** End of Disable Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables end of disable interrupt. |

**Bit 1 – LIE** End of Linked List Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables end of linked list interrupt. |

**Bit 0 – BIE** End of Block Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables end of block interrupt. |

### 37.9.19 XDMAC Channel x Interrupt Disable Register [x = 0..15]

**Name:** XDMAC_CID
**Offset:** 0x54 + n*0x40 [n=0..15]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| Access | | W | W | W | W | W | W | W |
| Reset | | – | – | – | – | – | – | – |

**Bit 6 – ROID** Request Overflow Error Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables request overflow error interrupt. |

**Bit 5 – WBEID** Write Bus Error Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables bus error interrupt. |

**Bit 4 – RBEID** Read Bus Error Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables bus error interrupt. |

**Bit 3 – FID** End of Flush Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables end of flush interrupt. |

**Bit 2 – DID** End of Disable Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables end of disable interrupt. |

**Bit 1 – LID** End of Linked List Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables end of linked list interrupt. |

**Bit 0 – BID** End of Block Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables end of block interrupt. |

### 37.9.20 XDMAC Channel x Interrupt Mask Register [x = 0..15]

**Name:** XDMAC_CIM
**Offset:** 0x58 + n*0x40 [n=0..15]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 6 – ROIM** Request Overflow Error Interrupt Mask

| Value | Description |
|---|---|
| 0 | Request overflow interrupt is masked. |
| 1 | Request overflow interrupt is activated. |

**Bit 5 – WBEIM** Write Bus Error Interrupt Mask

| Value | Description |
|---|---|
| 0 | Bus error interrupt is masked. |
| 1 | Bus error interrupt is activated. |

**Bit 4 – RBEIM** Read Bus Error Interrupt Mask

| Value | Description |
|---|---|
| 0 | Bus error interrupt is masked. |
| 1 | Bus error interrupt is activated. |

**Bit 3 – FIM** End of Flush Interrupt Mask

| Value | Description |
|---|---|
| 0 | End of flush interrupt is masked. |
| 1 | End of flush interrupt is activated. |

**Bit 2 – DIM** End of Disable Interrupt Mask

| Value | Description |
|---|---|
| 0 | End of disable interrupt is masked. |
| 1 | End of disable interrupt is activated. |

**Bit 1 – LIM** End of Linked List Interrupt Mask

| Value | Description |
|---|---|
| 0 | End of linked list interrupt is masked. |
| 1 | End of linked list interrupt is activated. |

**Bit 0 – BIM** End of Block Interrupt Mask

| Value | Description |
|---|---|
| 0 | Block interrupt is masked. |
| 1 | Block interrupt is activated. |

### 37.9.21 XDMAC Channel x Interrupt Status Register [x = 0..15]

**Name:** XDMAC_CIS
**Offset:** 0x5C + n*0x40 [n=0..15]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 6 – ROIS** Request Overflow Error Interrupt Status

| Value | Description |
|---|---|
| 0 | Overflow condition has not occurred. |
| 1 | Overflow condition has occurred at least once. (This information is only relevant for peripheral synchronized transfers.) |

**Bit 5 – WBEIS** Write Bus Error Interrupt Status

| Value | Description |
|---|---|
| 0 | Write bus error condition has not occurred. |
| 1 | At least one bus error has been detected in a write access since the last read of the Status register. |

**Bit 4 – RBEIS** Read Bus Error Interrupt Status

| Value | Description |
|---|---|
| 0 | Read bus error condition has not occurred. |
| 1 | At least one bus error has been detected in a read access since the last read of the Status register. |

**Bit 3 – FIS** End of Flush Interrupt Status

| Value | Description |
|---|---|
| 0 | End of flush condition has not occurred. |
| 1 | End of flush condition has occurred since the last read of the Status register. |

**Bit 2 – DIS** End of Disable Interrupt Status

| Value | Description |
|---|---|
| 0 | End of disable condition has not occurred. |
| 1 | End of disable condition has occurred since the last read of the Status register. |

**Bit 1 – LIS** End of Linked List Interrupt Status

| Value | Description |
|---|---|
| 0 | End of linked list condition has not occurred. |
| 1 | End of linked list condition has occurred since the last read of the Status register. |

**Bit 0 – BIS** End of Block Interrupt Status

| Value | Description |
|---|---|
| 0 | End of block interrupt has not occurred. |
| 1 | End of block interrupt has occurred since the last read of the Status register. |

### 37.9.22 XDMAC Channel x Source Address Register [x = 0..15]

**Name:** XDMAC_CSA
**Offset:** 0x60 + n*0x40 [n=0..15]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SA[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SA[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – SA[31:0]** Channel x Source Address
Program this register with the source address of the DMA transfer.

### 37.9.23 XDMAC Channel x Destination Address Register [x = 0..15]

**Name:** XDMAC_CDA
**Offset:** 0x64 + n*0x40 [n=0..15]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DA[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DA[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DA[31:0]** Channel x Destination Address
Program this register with the destination address of the DMA transfer.

### 37.9.24 XDMAC Channel x Next Descriptor Address Register [x = 0..15]

**Name:** XDMAC_CNDA
**Offset:** 0x68 + n*0x40 [n=0..15]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NDA[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NDA[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NDA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NDA[5:0] | | | | | | | NDAIF |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | 0 |

**Bits 31:2 – NDA[29:0]** Channel x Next Descriptor Address
The 30-bit width of the NDA field represents the next descriptor address range 31:2. The descriptor is word-aligned and the two least significant register bits 1:0 are ignored.

**Bit 0 – NDAIF** Channel x Next Descriptor Interface

| Value | Description |
|---|---|
| 0 | The channel descriptor is retrieved through system interface 0. |
| 1 | The channel descriptor is retrieved through system interface 1. |

### 37.9.25 XDMAC Channel x Next Descriptor Control Register [x = 0..15]

**Name:** XDMAC_CNDC
**Offset:** 0x6C + n*0x40 [n=0..15]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 4:3 – NDVIEW[1:0]** Channel x Next Descriptor View

| Value | Name | Description |
|---|---|---|
| 0 | NDV0 | Next Descriptor View 0 |
| 1 | NDV1 | Next Descriptor View 1 |
| 2 | NDV2 | Next Descriptor View 2 |
| 3 | NDV3 | Next Descriptor View 3 |

**Bit 2 – NDDUP** Channel x Next Descriptor Destination Update
0 (DST_PARAMS_UNCHANGED): Destination parameters remain unchanged.

1 (DST_PARAMS_UPDATED): Destination parameters are updated when the descriptor is retrieved.

**Bit 1 – NDSUP** Channel x Next Descriptor Source Update
0 (SRC_PARAMS_UNCHANGED): Source parameters remain unchanged.

1 (SRC_PARAMS_UPDATED): Source parameters are updated when the descriptor is retrieved.

**Bit 0 – NDE** Channel x Next Descriptor Enable
0 (DSCR_FETCH_DIS): Descriptor fetch is disabled.

1 (DSCR_FETCH_EN): Descriptor fetch is enabled.

### 37.9.26 XDMAC Channel x Microblock Control Register [x = 0..15]

**Name:** XDMAC_CUBC
**Offset:** 0x70 + n*0x40 [n=0..15]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UBLEN[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UBLEN[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UBLEN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – UBLEN[23:0]** Channel x Microblock Length
This field indicates the number of data in the microblock. The microblock contains UBLEN data.

### 37.9.27 XDMAC Channel x Block Control Register [x = 0..15]

**Name:** XDMAC_CBC
**Offset:** 0x74 + n*0x40 [n=0..15]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | BLEN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BLEN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 11:0 – BLEN[11:0]** Channel x Block Length
The length of the block is (BLEN+1) microblocks.

### 37.9.28 XDMAC Channel x Configuration Register [x = 0..15]

**Name:** XDMAC_CC
**Offset:** 0x78 + n*0x40 [n=0..15]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | PERID[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| Access | R/W | R/W | R/W | | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MEMSET | SWREQ | PROT | DSYNC | | MBSIZE[1:0] | | TYPE |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bits 30:24 – PERID[6:0]** Channel x Peripheral Hardware Request Line Identifier
This field contains the peripheral hardware request line identifier. PERID refers to identifiers defined in "DMA Controller Peripheral Connections".

**Note:** When a memory-to-memory transfer is performed, configure PERID to an unused peripheral ID (refer to table "Peripheral Identifiers").

**Bit 23 – WRIP** Write in Progress (this bit is read-only)
0 (DONE): No active write transaction on the bus.

1 (IN_PROGRESS): A write transaction is in progress.

**Bit 22 – RDIP** Read in Progress (this bit is read-only)
0 (DONE): No active read transaction on the bus.

1 (IN_PROGRESS): A read transaction is in progress.

**Bit 21 – INITD** Channel Initialization Done (this bit is read-only)
0 (IN_PROGRESS): Channel initialization is in progress.

1 (TERMINATED): Channel initialization is completed.

Note: When set to 0, XDMAC_CUBC.UBLEN and XDMAC_CNDA.NDA field values are unreliable each time a descriptor is being updated. See 37.8 XDMAC Software Requirements.

**Bits 19:18 – DAM[1:0]** Channel x Destination Addressing Mode

| Value | Name | Description |
|---|---|---|
| 0 | FIXED_AM | The address remains unchanged. |
| 1 | INCREMENTED_AM | The addressing mode is incremented (the increment size is set to the data size). |
| 2 | UBS_AM | The microblock stride is added at the microblock boundary. |
| 3 | UBS_DS_AM | The microblock stride is added at the microblock boundary; the data stride is added at the data boundary. |

**Bits 17:16 – SAM[1:0]** Channel x Source Addressing Mode

| Value | Name | Description |
|---|---|---|
| 0 | FIXED_AM | The address remains unchanged. |
| 1 | INCREMENTED_AM | The addressing mode is incremented (the increment size is set to the data size). |
| 2 | UBS_AM | The microblock stride is added at the microblock boundary. |
| 3 | UBS_DS_AM | The microblock stride is added at the microblock boundary, the data stride is added at the data boundary. |

**Bit 14 – DIF** Channel x Destination Interface Identifier
0 (AHB_IF0): The data is written through system bus interface 0.

1 (AHB_IF1): The data is written though system bus interface 1.

**Bit 13 – SIF** Channel x Source Interface Identifier
0 (AHB_IF0): The data is read through system bus interface 0.

1 (AHB_IF1): The data is read through system bus interface 1.

**Bits 12:11 – DWIDTH[1:0]** Channel x Data Width

| Value | Name | Description |
|---|---|---|
| 0 | BYTE | The data size is set to 8 bits |
| 1 | HALFWORD | The data size is set to 16 bits |
| 2 | WORD | The data size is set to 32 bits |
| 3 | DWORD | The data size is set to 64 bits |

**Bits 10:8 – CSIZE[2:0]** Channel x Chunk Size

| Value | Name | Description |
|---|---|---|
| 0 | CHK_1 | 1 data transferred |
| 1 | CHK_2 | 2 data transferred |
| 2 | CHK_4 | 4 data transferred |
| 3 | CHK_8 | 8 data transferred |
| 4 | CHK_16 | 16 data transferred |

**Bit 7 – MEMSET** Channel x Fill Block of Memory
0 (NORMAL_MODE): Memset is not activated.

1 (HW_MODE): Sets the block of memory pointed by DA field to the specified value. This operation is performed on 8-, 16- or 32-bit basis.

**Bit 6 – SWREQ** Channel x Software Request Trigger
0 (HWR_CONNECTED): Hardware request line is connected to the peripheral request line.

1 (SWR_CONNECTED): Software request is connected to the peripheral request line.

**Bit 5 – PROT** Channel x Protection
When a descriptor is loaded, the PROT bit cannot be modified. If PROT=0 for a channel, the configuration and status registers of this channel cannot be modified by unsecure software.
0 (SEC): Channel is secured.

1 (UNSEC): Channel is unsecured.

**Bit 4 – DSYNC** Channel x Synchronization
0 (PER2MEM): Peripheral-to-memory transfer.

1 (MEM2PER): Memory-to-peripheral transfer.

**Bits 2:1 – MBSIZE[1:0]** Channel x Memory Burst Size

| Value | Name | Description |
|---|---|---|
| 0 | SINGLE | The memory burst size is set to one. |
| 1 | FOUR | The memory burst size is set to four. |
| 2 | EIGHT | The memory burst size is set to eight. |
| 3 | SIXTEEN | The memory burst size is set to sixteen. |

**Bit 0 – TYPE** Channel x Transfer Type
0 (MEM_TRAN): Self-triggered mode (memory-to-memory transfer).

1 (PER_TRAN): Synchronized mode (peripheral-to-memory or memory-to-peripheral transfer).

### 37.9.29 XDMAC Channel x Data Stride Memory Set Pattern Register [x = 0..15]

**Name:** XDMAC_CDS_MSP
**Offset:** 0x7C + n*0x40 [n=0..15]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DDS_MSP[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DDS_MSP[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SDS_MSP[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SDS_MSP[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – DDS_MSP[15:0]** Channel x Destination Data Stride or Memory Set Pattern
When XDMAC_CCx.MEMSET = 0, this field indicates the destination data stride.

Number of bytes for the data stride of channel x (two's complement). If the field is set to zero the data is contiguous (see Data Striding Diagram).

The DDS_MSP field is only relevant when XDMAC_CCx.SAM=UBS_DS_AM.

When XDMAC_CCx.MEMSET = 1, this field indicates the memory set pattern.

**Bits 15:0 – SDS_MSP[15:0]** Channel x Source Data Stride or Memory Set Pattern
When XDMAC_CCx.MEMSET = 0, this field indicates the source data stride.

Number of bytes for the data stride of channel x (two's complement). If the field is set to zero the data is contiguous (see Data Striding Diagram).

The SDS_MSP field is only relevant when XDMAC_CCx.SAM=UBS_DS_AM.

When XDMAC_CCx.MEMSET = 1, this field indicates the memory set pattern.

### 37.9.30 XDMAC Channel x Source Microblock Stride Register [x = 0..15]

**Name:** XDMAC_CSUS
**Offset:** 0x80 + n*0x40 [n=0..15]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SUBS[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SUBS[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SUBS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – SUBS[23:0]** Channel x Source Microblock Stride
Number of bytes for the microblock stride for channel x (two's complement). If the field is set to zero the data is contiguous (see Figure 37-2).

The SUBS field is only relevant when XDMAC_CCx.SAM=UBS_AM or XDMAC_CCx.SAM=UBS_DS_AM.

### 37.9.31 XDMAC Channel x Destination Microblock Stride Register [x = 0..15]

**Name:** XDMAC_CDUS
**Offset:** 0x84 + n*0x40 [n=0..15]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DUBS[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DUBS[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DUBS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – DUBS[23:0]** Channel x Destination Microblock Stride
Number of bytes for the microblock stride for channel x (two's complement). If the field is set to zero the data is contiguous (see Figure 37-2).

The DUBS field is only relevant when XDMAC_CCx.SAM=UBS_AM or XDMAC_CCx.SAM=UBS_DS_AM.

# 38. LCD Controller (LCDC)

## 38.1 Description

The LCD Controller (LCDC) consists of logic for transferring LCD image data from an external display buffer to an LCD module. The LCD has one display input buffer per overlay that fetches pixels through the dual AHB master interface and a lookup table to allow palletized display configurations. The LCD controller is programmable on a per overlay basis, and supports different LCD resolutions, window sizes, image formats and pixel depths.

The LCD is connected to the Arm Advanced High Performance Bus (AHB) as a master for reading pixel data. It also integrates an APB interface to configure its registers.

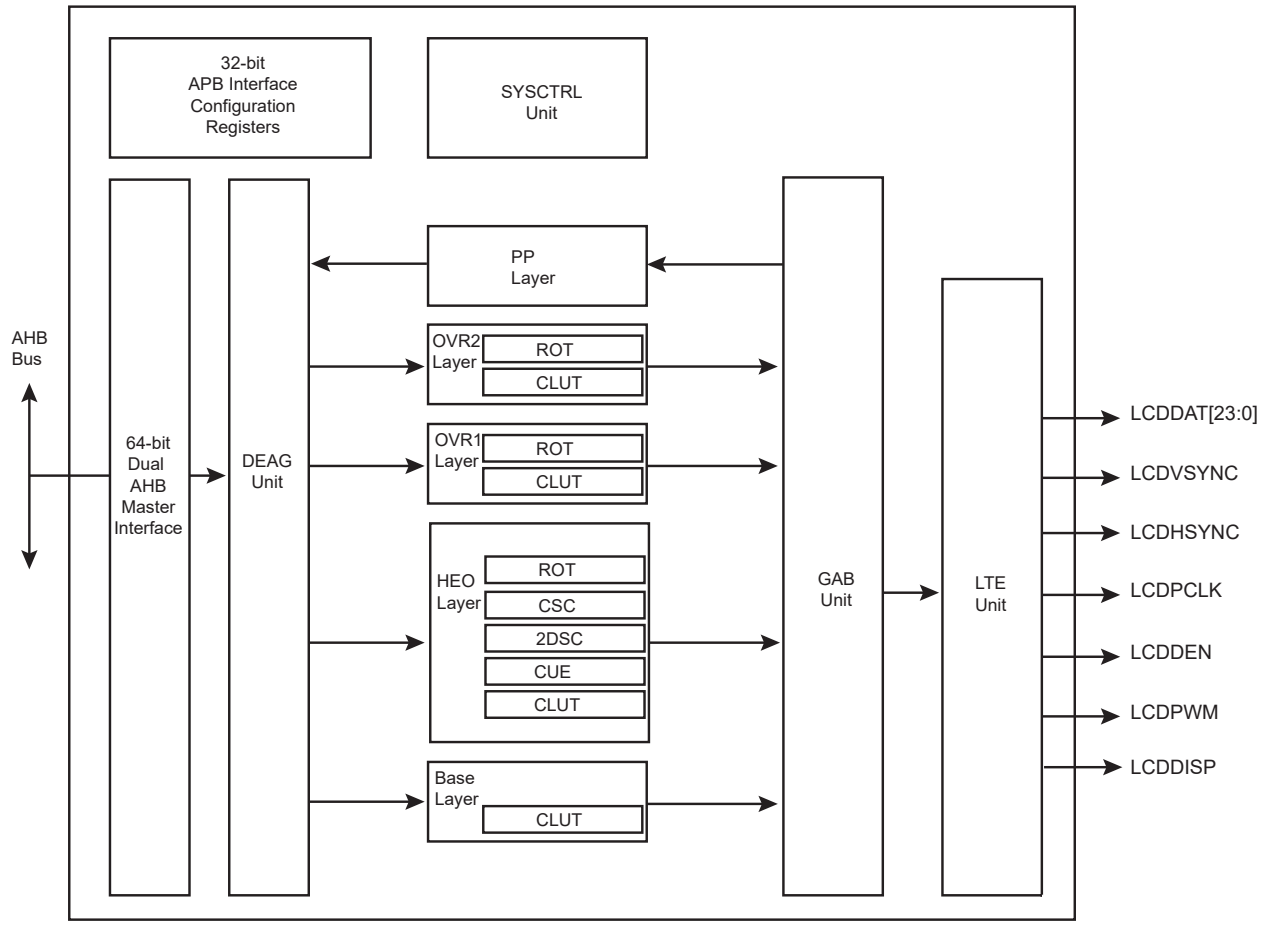
## 38.2 Embedded Characteristics

- Dual AHB Master Interface
- Supports Single Scan Active TFT Display
- Supports 12-, 16-, 18- and 24-bit Output Mode through the Spatial Dithering Unit
- Asynchronous Output Mode Supported
- 1, 2, 4, 8 bits per Pixel (Palletized)
- 12, 16, 18, 19, 24, 25 and 32 bits per Pixel (Non-palletized)
- Supports One Base Layer (Background)
- Supports One Overlay 1 Layer Window
- Supports One Overlay 2 Layer Window
- Supports One High-End Overlay (HEO) Window
- Little Endian Memory Organization
- Programmable Timing Engine, with Integer Clock Divider
- Programmable Polarity for Data, Line Synchro and Frame Synchro
- Up to 1024x768 (XGA) with Overlay (Application-Dependent). Still Image up to WXGA.
- Color Lookup Table with up to 256 Entries and Predefined 8-bit Alpha
- Programmable Negative and Positive Row Striding for all Layers
- Programmable Negative and Positive Pixel Striding for Layers
- High-End Overlay supports 4:2:0 Planar Mode and Semiplanar Mode
- High-End Overlay supports 4:2:2 Planar Mode, Semiplanar Mode and Packed
- High-End Overlay includes Chroma Upsampling Unit
- Horizontal and Vertical Rescaling Unit with Edge Interpolation and Independent Non-Integer Ratio, up to 1024x768
- Hidden Layer Removal supported
- Integrates Fully Programmable Color Space Conversion
- Blender Function Supports Arbitrary 8-bit Alpha Value and Chroma Keying
- DMA User Interface uses Linked List Structure and Add-to-queue Structure

© 2018 Microchip Technology Inc.

## 38.3 Block Diagram

**Figure 38-1. Block Diagram**




HEO: High-End Overlay
CUE: Chroma Upsampling Engine
CSC: Color Space Conversion
2DSC: Two-Dimension Scaler
DEAG: DMA Engine Address Generation
HCC: Hardware Cursor Channel
GAB: Global Alpha Blender
LTE: LCD Timing Engine
ROT: Hardware Rotation
OVRx: Overlay

## 38.4 I/O Lines Description

**Table 38-1. I/O Lines Description**

| Name | Description | Type |
|---|---|---|
| LCDPWM | Contrast control signal, using Pulse Width Modulation | Output |
| LCDHSYNC | Horizontal Synchronization Pulse | Output |
| LCDVSYNC | Vertical Synchronization Pulse | Output |
| LCDDAT[23:0] | LCD 24-bit data bus | Output |
| LCDDEN | Data Enable | Output |

| .....continued | | |
|---|---|---|
| **Name** | **Description** | **Type** |
| LCDDISP | Display Enable signal | Output |
| LCDPCLK | Pixel Clock | Output |

## 38.5 Product Dependencies

### 38.5.1 I/O Lines

The pins used for interfacing the LCD Controller may be multiplexed with PIO lines. The programmer must first program the PIO Controller to assign the pins to their peripheral function. If I/O lines of the LCD Controller are not used by the application, they can be used for other purposes by the PIO Controller.

### 38.5.2 Power Management

The LCD Controller is not continuously clocked. The user must first enable the LCD Controller clock in the Power Management Controller (PMC_PCER) before using it.

### 38.5.3 Interrupt Sources

The LCD Controller interrupt line is connected to one of the internal sources of the interrupt controller. Using the LCD Controller interrupt requires prior programming of the interrupt controller.

## 38.6 Functional Description

The LCD module integrates the following digital blocks:

- DMA Engine Address Generation (DEAG)—this block performs data prefetch and requests access to the AHB interface.
- Input Overlay FIFO—stores the stream of pixels
- Color Lookup Table (CLUT)—these 256 RAM-based lookup table entries are selected when the color depth is set to 1, 2, 4 or 8 bpp.
- Chroma Upsampling Engine (CUE)—this block is selected when the input image sampling format is YUV (Y'CbCr) 4:2:0 and converts it to higher quality 4:4:4 image.
- Color Space Conversion (CSC)—changes the color space from YUV to RGB
- Two Dimension Scaler (2DSC)—resizes the image
- Global Alpha Blender (GAB)—performs programmable 256-level alpha blending
- Output FIFO—stores the blended pixel prior to display
- LCD Timing Engine—provides a fully programmable HSYNC-VSYNC interface

The DMA controller reads the image through the AHB master interface. The LCD controller engine formats the display data, then the GAB performs alpha blending if required, and writes the final pixel into the output FIFO. The programmable timing engine drives a valid pixel onto the LCDDAT[23:0] display bus.

### 38.6.1 Timing Engine Configuration

#### 38.6.1.1 Pixel Clock Period Configuration

The pixel clock (LCDPCLK) generated by the timing engine is the source clock divided by the field CLKDIV in the LCDC_LCDCFG0 register. The source clock can be selected between the system clock and the 2x system clock with the field CLKSEL located in the LCDC_LCDCFG0 register.

Pixel clock period formula:

$$\text{LCD_PCLK} = \frac{\text{source clock}}{\text{CLKDIV}+2}$$

The pixel clock polarity is also programmable.

#### 38.6.1.2 Horizontal and Vertical Synchronization Configuration

The following fields are used to configure the timing engine:

- LCDC_LCDCFG1.HSPW
- LCDC_LCDCFG1.VSPW
- LCDC_LCDCFG2.VFPW
- LCDC_LCDCFG2.VBPW
- LCDC_LCDCFG3.HFPW
- LCDC_LCDCFG3.HBPW
- LCDC_LCDCFG4.PPL
- LCDC_LCDCFG4.RPF

The polarity of output signals is also programmable.

#### 38.6.1.3 Timing Engine Powerup Software Operation

The following sequence is used to enable the display:

1. Configure LCD timing parameters, signal polarity and clock period.
2. Enable the pixel clock by writing a one to bit LCDC_LCDEN.CLKEN.
3. Poll bit LCDC_LCDSR.CLKSTS to check that the clock is running.
4. Enable Horizontal and Vertical Synchronization by writing a one to bit LCDC_LCDEN.SYNCEN.
5. Poll bit LCDC_LCDSR.LCDSTS to check that the synchronization is up.
6. Enable the display power signal by writing a one to bit LCDC_LCDEN.DISPEN.
7. Poll bit LCDC_LCDSR.DISPSTS to check that the power signal is activated.

The field LCDC_LCDCFG5.GUARDTIME is used to configure the number of frames before the assertion of the DISP signal.

#### 38.6.1.4 Timing Engine Powerdown Software Operation

The following sequence is used to disable the display:

1. Disable the DISP signal by writing bit LCDC_LCDDIS.DISPDIS.
2. Poll bit LCDC_LCDSR.DISPSTS to verify that the DISP is no longer activated.
3. Disable the HSYNC and VSYNC signals by writing a one to bit LCDC_LCDDIS.SYNCDIS.
4. Poll bit LCDC_LCDSR.LCDSTS to check that the synchronization is off.
5. Disable the pixel clock by writing a one to bit LCDC_LCDDIS.CLKDIS.

### 38.6.2 DMA Software Operations

#### 38.6.2.1 DMA Channel Descriptor (DSCR) Alignment and Structure

The DMA Channel Descriptor (DSCR) must be aligned on a 64-bit boundary.

The DMA Channel Descriptor structure contains three fields:

- DSCR.CHXADDR: Frame Buffer base address register
- DSCR.CHXCTRL: Transfer Control register

- DSCR.CHXNEXT: Next Descriptor Address register

**Table 38-2. DMA Channel Descriptor Structure**

| System Memory | Structure Field for Channel CHX |
|---|---|
| DSCR + 0x0 | ADDR |
| DSCR + 0x4 | CTRL |
| DSCR + 0x8 | NEXT |

#### 38.6.2.2 Enabling a DMA Channel

Follow the steps below to enable a DMA channel:

1. Check the status of the channel by reading the CHXCHSR register.
2. Write the channel descriptor (DSCR) structure in the system memory by writing DSCR.CHXADDR Frame base address, DSCR.CHXCTRL channel control and DSCR.CHXNEXT next descriptor location.
3. If more than one descriptor is expected, the field DFETCH of DSCR.CHXCTRL is set to ‘1’ to enable the descriptor fetch operation.
4. Write the DSCR.CHXNEXT register with the address location of the descriptor structure and set DFETCH field of the DSCR.CHXCTRL register to ‘1’.
5. Enable the relevant channel by writing one to the CHEN field of the CHXCHER register.
6. An interrupt may be raised if unmasked when the descriptor has been loaded.

#### 38.6.2.3 Disabling a DMA Channel

Follow the steps below to disable a DMA channel:

1. Clearing the DFETCH bit in the DSCR.CHXCTRL field of the DSCR structure disables the channel at the end of the frame.
2. Setting the DSCR.CHXNEXT field of the DSCR structure disables the channel at the end of the frame.
3. Writing one to the CHDIS field of the CHXCHDR register disables the channel at the end of the frame.
4. Writing one to the CHRST field of the CHXCHDR register disables the channel immediately. This may occur in the middle of the image.
5. Polling CHSR field in the CHXCHSR register until the channel is successfully disabled.

#### 38.6.2.4 DMA Dynamic Linking of a New Transfer Descriptor

1. Write the new descriptor structure in the system memory.
2. Write the address of the new structure in the CHXHEAD register.
3. Add the new structure to the queue of descriptors by writing one to the A2QEN field of the CHXCHER register.
4. The new descriptor is added to the queue on the next frame.
5. An interrupt is raised if unmasked, when the head descriptor structure has been loaded by the DMA channel.

#### 38.6.2.5 DMA Interrupt Generation

The DMA Controller operation sets the following interrupt flags in the Interrupt Status register CHXISR:

- DMA field indicates that the DMA transfer is completed.
- DSCR field indicates that the descriptor structure is loaded in the DMA controller.

- ADD field indicates that a descriptor has been added to the descriptor queue.
- DONE field indicates that the channel transfer has terminated and the channel is automatically disabled.

#### 38.6.2.6 DMA Address Alignment Requirements

When programming the DSCR.CHXADDR field of the DSCR structure, the following requirement must be met.

**Table 38-3. DMA Address Alignment when CLUT Mode is Selected**

| CLUT Mode | DMA Address Alignment |
|---|---|
| 1 bpp | 8 bits |
| 2 bpp | 8 bits |
| 4 bpp | 8 bits |
| 8 bpp | 8 bits |

**Table 38-4. DMA Address Alignment when RGB Mode is Selected**

| RGB Mode | DMA Address Alignment |
|---|---|
| 12 bpp RGB 444 | 16 bits |
| 16 bpp ARGB 4444 | 16 bits |
| 16 bpp RGBA 4444 | 16 bits |
| 16 bpp RGB 565 | 16 bits |
| 16 bpp TRGB 1555 | 16 bits |
| 18 bpp RGB 666 | 32 bits |
| 18 bpp RGB 666 PACKED | 8 bits |
| 19 bpp TRGB 1666 | 32 bits |
| 19 bpp TRGB 1666 | 8 bits |
| 24 bpp RGB 888 | 32 bits |
| 24 bpp RGB 888 PACKED | 8 bits |
| 25 bpp TRGB 1888 | 32 bits |
| 32 bpp ARGB 8888 | 32 bits |
| 32 bpp RGBA 8888 | 32 bits |

**Table 38-5. DMA Address Alignment when YUV Mode is Selected**

| YUV Mode | DMA Address Alignment |
|---|---|
| 32 bpp AYCrCb | 32 bits |
| 16 bpp YCrCb 4:2:2 | 32 bits |

| .....continued | |
|---|---|
| **YUV Mode** | **DMA Address Alignment** |
| 16 bpp semiplanar YCrCb 4:2:2 | Y 8 bits |
| | CrCb 16 bits |
| 16 bpp planar YCrCb 4:2:2 | Y 8 bits |
| | Cr 8 bits |
| | Cb 8 bits |
| 12 bpp YCrCb 4:2:0 | Y 8 bits |
| | CrCb 16 bits |
| 12 bpp YCrCb 4:2:0 | Y 8 bits |
| | Cr 8 bits |
| | Cb 8 bits |

### 38.6.3 Overlay Software Configuration

#### 38.6.3.1 System Bus Access Attributes

These attributes are defined to improve bandwidth of the overlay.

- LOCKDIS bit—When set to ‘1’, the AHB lock signal is not asserted when the PSTRIDE value is different from zero (rotation in progress).
- ROTDIS bit—When set to ‘1’, the Pixel Striding optimization is disabled.
- DLBO bit—When set to ‘1’, only defined burst lengths are performed when the DMA channel retrieves the data from the memory.
- BLEN field—Defines the maximum burst length of the DMA channel.
- SIF bit—Defines the targeted DMA interface.

#### 38.6.3.2 Color Attributes

- CLUTMODE field—Selects the Color Lookup Table mode.
- RGBMODE field—Selects the RGB mode.
- YUVMODE field—Selects the Luminance Chrominance mode.

#### 38.6.3.3 Window Position, Size, Scaling and Striding Attributes

- XPOS and YPOS fields—Defines the position of the overlay window.
- XSIZE and YSIZE fields—Defines the size of the displayed window.
- XMEMSIZE and YMEMSIZE fields—Defines the size of the image frame buffer.
- XSTRIDE and PSTRIDE fields—Defines the line and pixel striding.
- XFACTOR and YFACTOR fields—Defines the scaling ratio.

The position and size attributes are to be programmed to keep the window within the display area.

When the Color Lookup Table mode is enabled, the restrictions detailed in the following table apply on the horizontal and vertical window sizes.

**Table 38-6. Color Lookup Table Mode and Window Size**

| CLUT Mode | X-Y Size Requirement |
|---|---|
| 1 bpp | Multiple of 8 pixels |
| 2 bpp | Multiple of 4 pixels |
| 4 bpp | Multiple of 2 pixels |
| 8 bpp | Free size |

Pixel striding is disabled when CLUT mode is enabled.

When YUV mode is enabled, the restrictions detailed in the following table apply on the window size.

**Table 38-7. YUV Mode and Window Size**

| YUV Mode | X-Y Requirement, Scaling Turned Off | X-Y Requirement, Scaling Turned On |
|---|---|---|
| AYUV | Free size | X-Y size is greater than 5 |
| YUV 4:2:2 packed | XSIZE is greater than 2 pixels | X-Y size is greater than 5 |
| YUV 4:2:2 semiplanar | XSIZE is greater than 2 pixels | X-Y size is greater than 5 |
| YUV 4:2:2 planar | XSIZE is greater than 2 pixels | X-Y size is greater than 5 |
| YUV 4:2:0 semiplanar | XSIZE is greater that 2 pixels | X-Y size is greater than 5 |
| YUV 4:2:0 planar | XSIZE is greater than 2 pixels | X-Y size is greater than 5 |

In RGB mode, there is no restriction on the line length.

#### 38.6.3.4 Overlay Blender Attributes

When two or more video layers are used, alpha blending is performed to define the final image displayed. Each window has its own blending attributes.

- CRKEY bit—Enables the chroma keying and match logic.
- INV bit—Performs bit inversion at pixel level.
- ITER2BL bit—When written to ‘1’, the iterated data path is selected.
- ITER bit—When written to ‘1’, the iterated value is used in the iterated data path, otherwise the iterated value is set to 0.
- REVALPHA bit—Uses the reverse alpha value.
- GAEN bit—Enables the global alpha value in the data path.
- LAEN bit—Enables the local alpha value from the pixel.
- OVR bit—When written to ‘1’, the overlay is selected as an input of the blender.
- DMA bit—The DMA data path is activated.
- REP bit—Enables the bit replication to fill the 24-bit internal data path.
- DSTKEY bit—When written to ‘1’, Destination keying is enabled.
- GA field—Defines the global alpha value.

#### 38.6.3.5 Overlay Attributes Software Operation

1. When required, write the overlay attributes configuration registers.
2. Set UPDATEEN field of the CHXCHER register.
3. Poll UPDATESR field in the CHXCHSR, the update applies when that field is reset.

### 38.6.4 RGB Frame Buffer Memory Bitmap

#### 38.6.4.1 1 bpp Through Color Lookup Table

**Table 38-8. 1 bpp Memory Mapping, Little Endian Organization**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 1 bpp | p31 | p30 | p29 | p28 | p27 | p26 | p25 | p24 | p23 | p22 | p21 | p20 | p19 | p18 | p17 | p16 | p15 | p14 | p13 | p12 | p11 | p10 | p9 | p8 | p7 | p6 | p5 | p4 | p3 | p2 | p1 | p0 |

#### 38.6.4.2 2 bpp Through Color Lookup Table

**Table 38-9. 2 bpp Memory Mapping, Little Endian Organization**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 2 bpp | p15 | | p14 | | p13 | | p12 | | p11 | | p10 | | p9 | | p8 | | p7 | | p6 | | p5 | | p4 | | p3 | | p2 | | p1 | | p0 | |

#### 38.6.4.3 4 bpp Through Color Lookup Table

**Table 38-10. 4 bpp Memory Mapping, Little Endian Organization**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 4 bpp | p7 | | | | p6 | | | | p5 | | | | p4 | | | | p3 | | | | p2 | | | | p1 | | | | p0 | | | |

#### 38.6.4.4 8 bpp Through Color Lookup Table

**Table 38-11. 8 bpp Memory Mapping, Little Endian Organization**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 8 bpp | p3 | | | | | | | | p2 | | | | | | | | p1 | | | | | | | | p0 | | | | | | | |

#### 38.6.4.5 12 bpp Memory Mapping, RGB 4:4:4

**Table 38-12. 12 bpp Memory Mapping, Little Endian Organization**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 12 bpp | - | | | | R1[3:0] | | | | G1[3:0] | | | | B1[3:0] | | | | - | | | | R0[3:0] | | | | G0[3:0] | | | | B0[3:0] | | | |

#### 38.6.4.6 16 bpp Memory Mapping with Alpha Channel, ARGB 4:4:4:4

**Table 38-13. 16 bpp Memory Mapping, Little Endian Organization**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 16 bpp | A1[3:0] | | | | R1[3:0] | | | | G1[3:0] | | | | B1[3:0] | | | | A0[3:0] | | | | R0[3:0] | | | | G0[3:0] | | | | B0[3:0] | | | |

#### 38.6.4.7 16 bpp Memory Mapping with Alpha Channel, RGBA 4:4:4:4

**Table 38-14. 16 bpp Memory Mapping, Little Endian Organization**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 16 bpp | R1[3:0] | | | | G13:0] | | | | B1[3:0] | | | | A1[3:0] | | | | R0[3:0] | | | | G0[3:0] | | | | B0[3:0] | | | | A0[3:0] | | | |

#### 38.6.4.8 16 bpp Memory Mapping with Alpha Channel, RGB 5:6:5

**Table 38-15. 16 bpp Memory Mapping, Little Endian Organization**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 16bpp | R1[4:0] | | | | | G1[5:0] | | | | | | B1[4:0] | | | | | R0[4:0] | | | | | G0[5:0] | | | | | | B0[4:0] | | | |

#### 38.6.4.9 16 bpp Memory Mapping with Transparency Bit, ARGB 1:5:5:5

**Table 38-16. 16 bpp Memory Mapping, Little Endian Organization**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 4 bpp | A1 | R1[4:0] | | | | | G1[4:0] | | | | | B1[4:0] | | | | | A0 | R0[4:0] | | | | | G0[4:0] | | | | | B0[4:0] | | | |

#### 38.6.4.10 18 bpp Unpacked Memory Mapping with Transparency Bit, RGB 6:6:6

**Table 38-17. 18 bpp Unpacked Memory Mapping, Little Endian Organization**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 18 bpp | – | | | | | | | | – | | | | | | R0[5:0] | | | | | | G0[5:0] | | | | | | B0[5:0] | | | |

### 38.6.4.11 18 bpp Packed Memory Mapping with Transparency Bit, RGB 6:6:6

**Table 38-18. 18 bpp Packed Memory Mapping, Little Endian Organization at Address 0x0, 0x1, 0x2, 0x3**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 18 bpp | G1[1:0] | | B1[5:0] | | | | | | – | | | | | | R0[5:0] | | | | | | G0[5:0] | | | | B0[5:0] | | | | | | | |

**Table 38-19. 18 bpp Packed Memory Mapping, Little Endian Organization at Address 0x4, 0x5, 0x6, 0x7**

| Mem addr | 0x7 | | | | | | | | 0x6 | | | | | | | | 0x5 | | | | | | | | 0x4 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 18 bpp | R2[3:0] | | | | G2[5:0] | | | | B2[5:0] | | | | | | | | – | | | | | | R1[5:2] | | | | | | G1[5:2] | | | |

**Table 38-20. 18 bpp Packed Memory Mapping, Little Endian Organization at Address 0x8, 0x9, 0xA, 0xB**

| Mem addr | 0xB | | | | | | | | 0xA | | | | | | | | 0x9 | | | | | | | | 0x8 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 18 bpp | G4[1:0] | | B4[5:0] | | | | | | – | | | | R3[5:0] | | | | G3[5:0] | | | | | | | | B3[3:0] | | | | | | R2[5:4] | |

### 38.6.4.12 19 bpp Unpacked Memory Mapping with Transparency Bit, RGB 1:6:6:6

**Table 38-21. 19 bpp Unpacked Memory Mapping, Little Endian Organization**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 19 bpp | – | | | | | | | | – | | | | | A0 | R0[5:0] | | | | | | G0[5:0] | | | | B0[5:0] | | | | | | | |

### 38.6.4.13 19 bpp Packed Memory Mapping with Transparency Bit, ARGB 1:6:6:6

**Table 38-22. 19 bpp Packed Memory Mapping, Little Endian Organization at Address 0x0, 0x1, 0x2, 0x3**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 19 bpp | G1[1:0] | | B1[5:0] | | | | | | – | | | | | A0 | R0[5:0] | | | | | | G0[5:0] | | | | B0[5:0] | | | | | | | |

**Table 38-23. 19 bpp Packed Memory Mapping, Little Endian Organization at Address 0x4, 0x5, 0x6, 0x7**

| Mem addr | 0x7 | | | | | | | | 0x6 | | | | | | | | 0x5 | | | | | | | | 0x4 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 19 bpp | R2[3:0] | | | | G2[5:0] | | | | B2[5:0] | | | | | | | | – | | | | | A1 | R1[5:2] | | | | | | G1[5:2] | | | |

**Table 38-24. 18 bpp Packed Memory Mapping, Little Endian Organization at Address 0x8, 0x9, 0xA, 0xB**

| Mem addr | 0xB | | | | | | | | 0xA | | | | | | | | 0x9 | | | | | | | | 0x8 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 19 bpp | G4[1:0] | | B4[5:0] | | | | | | – | | | A3 | R3[5:0] | | | | G3[5:0] | | | | | | | | B3[3:0] | | | | | | R2[5:4] | |

### 38.6.4.14 24 bpp Unpacked Memory Mapping, RGB 8:8:8

**Table 38-25. 24 bpp Memory Mapping, Little Endian Organization**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 24 bpp | – | | | | | | | | R0[7:0] | | | | | | | | G0[7:0] | | | | | | | | B0[7:0] | | | | | | | |

### 38.6.4.15 24 bpp Packed Memory Mapping, RGB 8:8:8

**Table 38-26. 24 bpp Packed Memory Mapping, Little Endian Organization at Address 0x0, 0x1, 0x2, 0x3**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 24 bpp | B1[7:0] | | | | | | | | R0[7:0] | | | | | | | | G0[7:0] | | | | | | | | B0[7:0] | | | | | | | |

**Table 38-27. 24 bpp Packed Memory Mapping, Little Endian Organization at Address 0x4, 0x5, 0x6, 0x7**

| Mem addr | 0x7 | | | | | | | | 0x6 | | | | | | | | 0x5 | | | | | | | | 0x4 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 24 bpp | G2[7:0] | | | | | | | | B2[7:0] | | | | | | | | R1[7:0] | | | | | | | | G1[7:0] | | | | | | | |

#### 38.6.4.16 25 bpp Memory Mapping, ARGB 1:8:8:8

**Table 38-28. 25 bpp Memory Mapping, Little Endian Organization**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 25 bpp | – | | | | | | | A0 | R0[7:0] | | | | | | | | G0[7:0] | | | | | | | | B0[7:0] | | | | | | | |

#### 38.6.4.17 32 bpp Memory Mapping, ARGB 8:8:8:8

**Table 38-29. 32 bpp Memory Mapping, Little Endian Organization**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 32 bpp | A0[7:0] | | | | | | | | R0[7:0] | | | | | | | | G0[7:0] | | | | | | | | B0[7:0] | | | | | | | |

#### 38.6.4.18 32 bpp Memory Mapping, RGBA 8:8:8:8

**Table 38-30. 32 bpp Memory Mapping, Little Endian Organization**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 32 bpp | R0[7:0] | | | | | | | | G0[7:0] | | | | | | | | B0[7:0] | | | | | | | | A0[7:0] | | | | | | | |

### 38.6.5 YUV Frame Buffer Memory Mapping

#### 38.6.5.1 AYCbCr 4:4:4 Interleaved Frame Buffer Memory Mapping

**Table 38-31. 32 bpp Memory Mapping, Little Endian Organization**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 16 bpp | A0[7:0] | | | | | | | | Y0[7:0] | | | | | | | | Cb0[7:0] | | | | | | | | Cr0[7:0] | | | | | | | |

#### 38.6.5.2 4:2:2 Interleaved Mode Frame Buffer Memory Mapping

**Table 38-32. 16 bpp 4:2:2 Interleaved Mode 0**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 16 bpp | Cr0[7:0] | | | | | | | | Y1[7:0] | | | | | | | | Cb0[7:0] | | | | | | | | Y0[7:0] | | | | | | | |

**Table 38-33. 16 bpp 4:2:2 Interleaved Mode 1**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 16 bpp | Y1[7:0] | | | | | | | | Cr0[7:0] | | | | | | | | Y0[7:0] | | | | | | | | Cb0[7:0] | | | | | | | |

**Table 38-34. 16 bpp 4:2:2 Interleaved Mode 2**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 16 bpp | Cb0[7:0] | | | | | | | | Y1[7:0] | | | | | | | | Cr0[7:0] | | | | | | | | Y0[7:0] | | | | | | | |

**Table 38-35. 16 bpp 4:2:2 Interleaved Mode 3**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 16 bpp | Y1[7:0] | | | | | | | | Cb0[7:0] | | | | | | | | Y0[7:0] | | | | | | | | Cr0[7:0] | | | | | | | |

### 38.6.5.3 4:2:2 Semiplanar Mode Frame Buffer Memory Mapping

**Table 38-36. 4:2:2 Semiplanar Luminance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 16 bpp | Y3[7:0] | | | | | | | | Y2[7:0] | | | | | | | | Y1[7:0] | | | | | | | | Y0[7:0] | | | | | | | |

**Table 38-37. 4:2:2 Semiplanar Chrominance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 16 bpp | Cb2[7:0] | | | | | | | | Cr2[7:0] | | | | | | | | Cb0[7:0] | | | | | | | | Cr0[7:0] | | | | | | | |

### 38.6.5.4 4:2:2 Planar Mode Frame Buffer Memory Mapping

**Table 38-38. 4:2:2 Planar Mode Luminance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 16 bpp | Y3[7:0] | | | | | | | | Y2[7:0] | | | | | | | | Y1[7:0] | | | | | | | | Y0[7:0] | | | | | | | |

**Table 38-39. 4:2:2 Planar Mode Chrominance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 16 bpp | C3[7:0] | | | | | | | | C2[7:0] | | | | | | | | C1[7:0] | | | | | | | | C0[7:0] | | | | | | | |

### 38.6.5.5 4:2:0 Planar Mode Frame Buffer Memory Mapping

In Planar mode, the three video components Y, Cr and Cb are split into three memory areas and stored in a raster-scan order. These three memory planes are contiguous and always aligned on a 32-bit boundary.

**Table 38-40. 4:2:0 Planar Mode Luminance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 12 bpp | Y3[7:0] | | | | | | | | Y2[7:0] | | | | | | | | Y1[7:0] | | | | | | | | Y0[7:0] | | | | | | | |

**Table 38-41. 4:2:0 Planar Mode Luminance Memory Mapping, Little Endian Organization for Byte 0x4, 0x5, 0x6, 0x7**

| Mem addr | 0x7 | | | | | | | | 0x6 | | | | | | | | 0x5 | | | | | | | | 0x4 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 12 bpp | Y7[7:0] | | | | | | | | Y6[7:0] | | | | | | | | Y5[7:0] | | | | | | | | Y4[7:0] | | | | | | | |

**Table 38-42. 4:2:0 Planar Mode Chrominance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

| .....continued | | | | |
|---|---|---|---|---|
| **Mem addr** | **0x3** | **0x2** | **0x1** | **0x0** |
| Pixel 12 bpp | C3[7:0] | C2[7:0] | C1[7:0] | C0[7:0] |

**Table 38-43. 4:2:0 Planar Mode Chrominance Memory Mapping, Little Endian Organization for Byte 0x4, 0x5, 0x6, 0x7**

| Mem addr | 0x7 | | | | | | | | 0x6 | | | | | | | | 0x5 | | | | | | | | 0x4 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 12 bpp | C7[7:0] | | | | | | | | C6:[7:0] | | | | | | | | C5[7:0] | | | | | | | | C4[7:0] | | | | | | | |

### 38.6.5.6 4:2:0 Semiplanar Frame Buffer Memory Mapping

**Table 38-44. 4:2:0 Semiplanar Mode Luminance Memory Mapping, Little Endian Organization**

| Mem addr | 0x7 | | | | | | | | 0x6 | | | | | | | | 0x5 | | | | | | | | 0x4 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 12 bpp | Y3[7:0] | | | | | | | | Y2[7:0] | | | | | | | | Y1[7:0] | | | | | | | | Y0[7:0] | | | | | | | |

**Table 38-45. 4:2:0 Semiplanar Mode Chrominance Memory Mapping, Little Endian Organization**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 12 bpp | Cb1[7:0] | | | | | | | | Cr1[7:0] | | | | | | | | Cb0[7:0] | | | | | | | | Cr0[7:0] | | | | | | | |

### 38.6.6 Chrominance Upsampling Unit

Both the 4:2:2 and the 4:2:0 input formats are supported by the LCD module. In 4:2:2, the two chrominance components are sampled at half the luminance sample rate. The horizontal chrominance resolution is halved. When this input format is selected, the chrominance upsampling unit uses two chrominances to interpolate the missing component.

In 4:2:0, Cr and Cb components are subsampled at a factor of two vertically and horizontally. When this input mode is selected, the chrominance upsampling unit uses two and four chroma components to generate the missing horizontal and vertical components.

**Figure 38-2. 4:2:2 Upsampling Algorithm**

Vertical and Horizontal upsampling 4:2:2 to 4:4:4 conversion 0 or 180 degrees

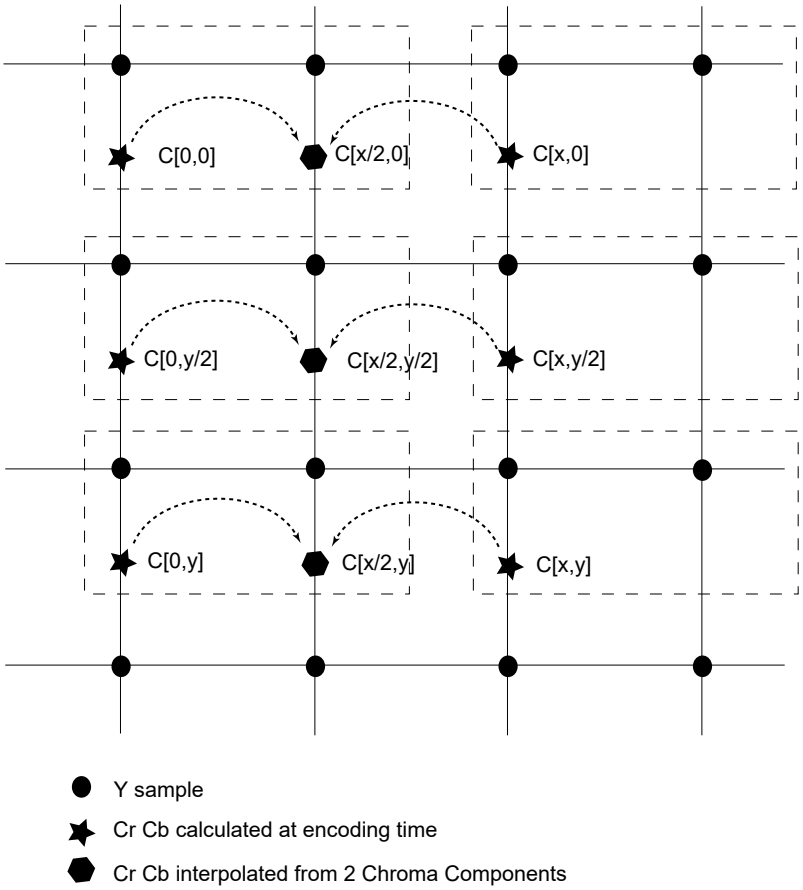


**Figure 38-3. 4:2:2 Packed Upsampling Algorithm**

Vertical and Horizontal upsampling 4:2:2 to 4:4:4 conversion 90 or 270 degrees

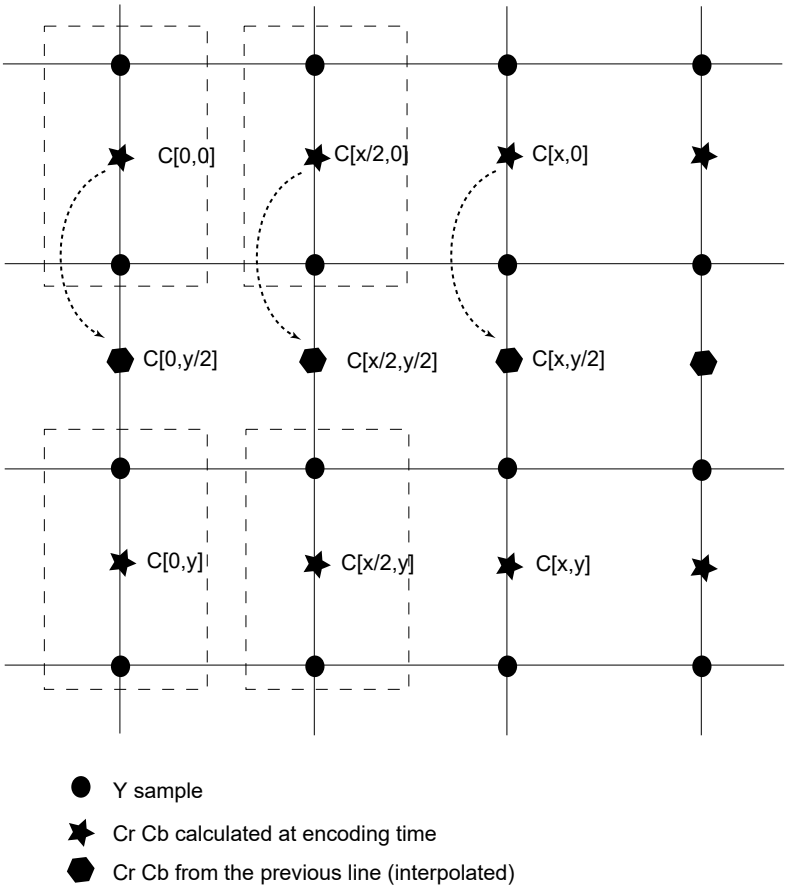


**Figure 38-4. 4:2:2 Semiplanar and Planar Upsampling Algorithm - 90 or 270 Degree R Rotation Activated**

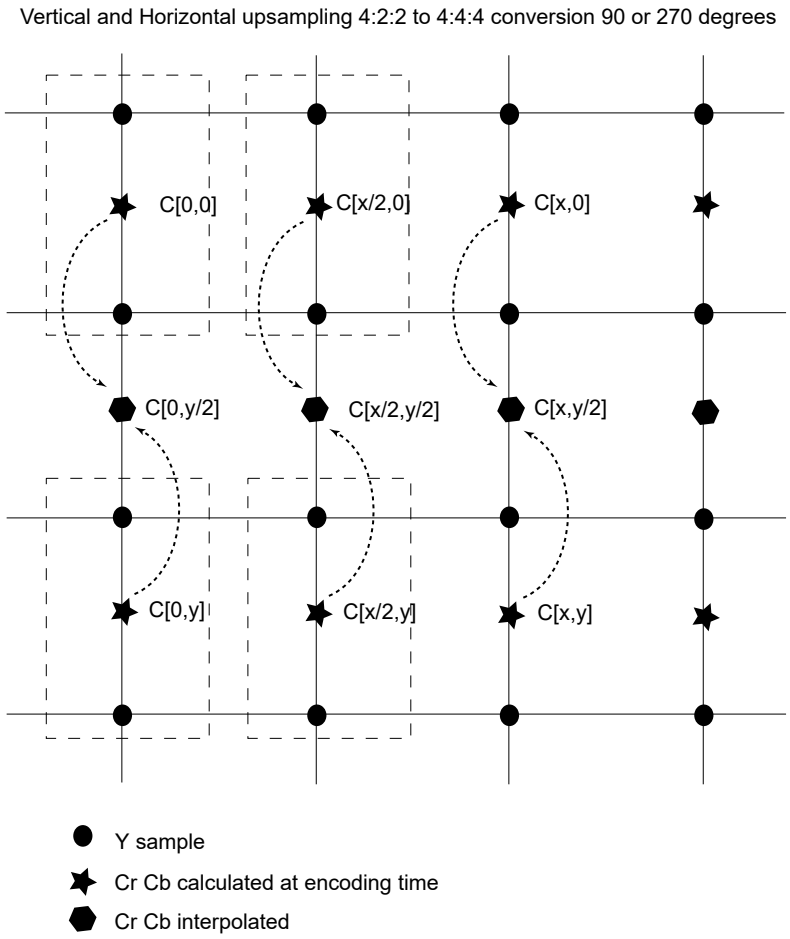
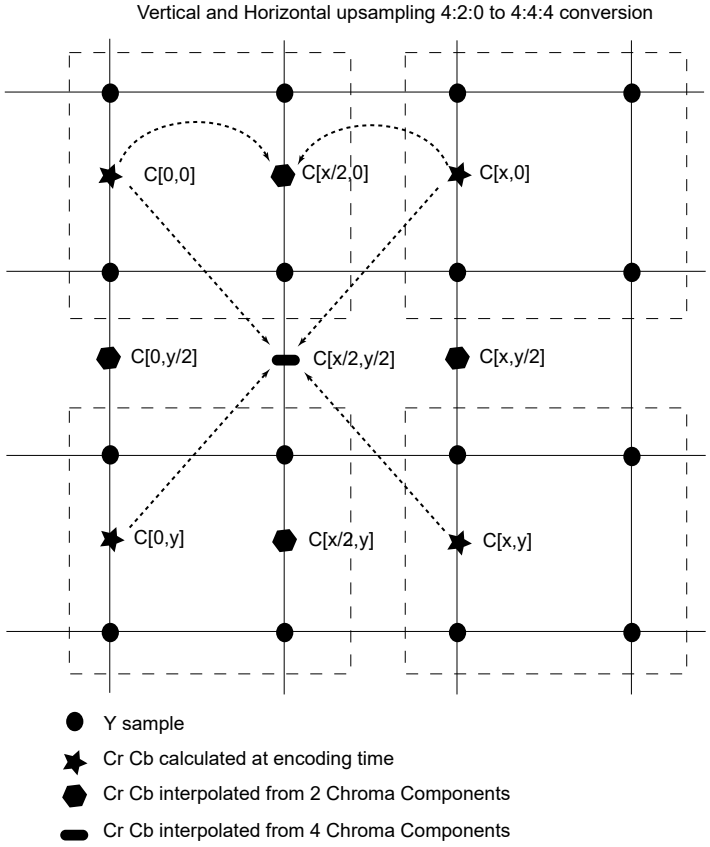


**Figure 38-5. 4:2:0 Upsampling Algorithm**




$$\text{Chroma}\left[\frac{x}{2}, 0\right] = \frac{\text{Cr}[0,0] + \text{Cr}[0,x]}{2}$$

$$\text{Chroma}\left[0, \frac{y}{2}\right] = \frac{\text{Cr}[0,0] + C[0,y]}{2}$$

$$\text{Chroma}\left[\frac{x}{2}, \frac{y}{2}\right] = \frac{\text{Cr}[0,0] + \text{Cr}[x,0] + \text{Cr}[y,0] + \text{Cr}[x,y]}{4}$$

$$\text{Chroma}\left[x, \frac{y}{2}\right] = \frac{\text{Cr}[x,0] + \text{Cr}[x,y]}{2}$$

$$\text{Chroma}\left[\frac{x}{2}, y\right] = \frac{\text{Cr}[0,y] + \text{Cr}[x,y]}{2}$$

#### 38.6.6.1 Chrominance Upsampling Algorithm

1. Read line n from chrominance cache and interpolate [x/2,0] chrominance component filling the 1 x 2 kernel with line n. If the chrominance cache is empty, then fetch the first line from external memory and interpolate from the external memory. Duplicate the last chrominance at the end of line.
2. Fetch line n+1 from external memory, write line n + 1 to chrominance cache, read line n from the chrominance cache. Interpolate [0,y/2], [x/2,y/2] and [x, y/2] filling the 2x2 kernel with lines n and n +1. Duplicate the last chrominance line to generate the last interpolated line.
3. Repeat step 1 and step 2.

### 38.6.7 Line and Pixel Striding

The LCD module includes a technique to increment the memory address by a programmable amount when the end of line has been reached. This offset is referred to as XSTRIDE and is defined on a per overlay basis. Additionally, the PSTRIDE field allows a programmable jump at the pixel level. Pixel stride is the value from one pixel to the next.

#### 38.6.7.1 Line Striding

When the end of line has been reached, the DMA address counter points to the next pixel address. The channel DMA address register is added to the XSTRIDE field, and then updated. If XSTRIDE is set to '0', the DMA address register remains unchanged. The XSTRIDE field of the channel configuration register is aligned to the pixel size boundary. The XSTRIDE field is a two's complement number. The following formula applies at the line boundary and indicates how the DMA controller computes the next pixel address. The function Sizeof() returns the number of bytes required to store a pixel.

$$\text{NextPixelAddress} = \text{CurrentPixelAddress} + \text{Sizeof}(\text{pixel}) + \text{XSTRIDE}$$

#### 38.6.7.2 Pixel Striding

The DMA channel engine may optionally fetch non-contiguous pixels. The channel DMA address register is added to the PSTRIDE field and then updated. If PSTRIDE is set to zero, the DMA address register remains unchanged and pixels are contiguous. The PSTRIDE field of the channel configuration register is aligned to the pixel size boundary. The PSTRIDE is a two's complement number. The following formula applies at the pixel boundary and indicates how the DMA controller computes the next pixel address. The function Sizeof() returns the number of bytes required to store a pixel.

$$\text{NextPixelAddress} = \text{CurrentPixelAddress} + \text{Sizeof}(\text{pixel}) + \text{PSTRIDE}$$

### 38.6.8 Color Space Conversion Unit

The color space conversion unit converts Luminance Chrominance color space into the Red Green Blue color space. The conversion matrix is defined below and is fully programmable through the LCD user interface.

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} \text{CSCRY} & \text{CSCRU} & \text{CSCRV} \\ \text{CSCGY} & \text{CSCGU} & \text{CSCGV} \\ \text{CSCBY} & \text{CSCBU} & \text{CSCBV} \end{bmatrix} \cdot \begin{bmatrix} Y - \text{Yoff} \\ \text{Cb} - \text{Cboff} \\ \text{Cr} - \text{Croff} \end{bmatrix}$$

Color space conversion coefficients are defined with the following equation:

$$\text{CSC}_{\text{ij}} = \frac{1}{2^7} \cdot \left[ -2^9 \cdot c_9 + \sum_{n=0}^{8} c_n \cdot 2^n \right]$$

Color space conversion coefficients are defined with one sign bit, 2 integer bits and 7 fractional bits. The range of the CSCij coefficients is defined below with a step of 1/128.

$$-4 \leq \text{CSC}_{\text{ij}} \leq 3.9921875$$

Additionally, a set scaling factor {Yoff, Cboff, Croff} can be applied.

### 38.6.9 Two-Dimension Scaler

The High-End Overlay (HEO) data path includes a hardware scaler that allows an image resize in both the horizontal and the vertical directions.

#### 38.6.9.1 Video Scaler Description

The scaling operation is based on a vertical and horizontal resampling algorithm. The sampling rate of the original image is increased when the video is upscaled, and decreased when the video is downscaled. A Vertical resampler is used to perform a vertical interpolation by a factor of vI, and a decimation by a factor

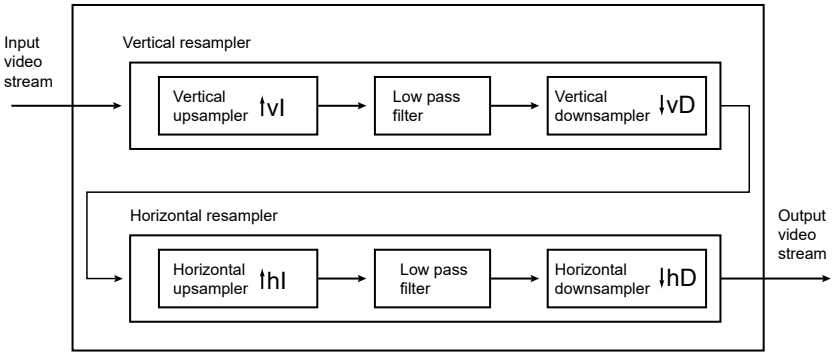
of vD. A Horizontal resampler is used to perform a vertical interpolation by a factor of hI, and a decimation by a factor of hD. The horizontal and vertical low pass filters are both designed to minimize the aliasing effect. The frequency response of the low pass filter has the following characteristics:

$$H(\omega) = \begin{cases} I & \text{when } 0 \le |\omega| \le \min\left(\frac{\pi}{I}, \frac{\pi}{D}\right) \\ 0 & \text{otherwise} \end{cases}$$

Taking into account the linear phase condition and anticipating the filter length M, the desired frequency response is modified.

$$H(\omega) = \begin{cases} Ie^{-j\omega\frac{M}{2}} & \text{when } 0 \le |\omega| \le \min\left(\frac{\pi}{I}, \frac{\pi}{D}\right) \\ 0 & \text{otherwise} \end{cases}$$

**Figure 38-6. Video Resampler Architecture**




The impulse response of the defined low pass filter is:

$$h(n) = \begin{cases} I \times \frac{\omega_c}{\pi} \text{ when } n = 0 \\ I \times \frac{\omega_c}{\pi} \times \frac{\sin\omega cn}{\omega_c n} \text{otherwise} \end{cases}$$

Or, for the filter of length M:

$$h(n) = \begin{cases} I \times \frac{\omega_c}{\pi} \text{ when } n = \frac{M}{2} \\ I \times \frac{\omega_c}{\pi} \times \frac{\sin\omega cn - M2}{\omega_c\left(n - \frac{M}{2}\right)} \text{otherwise} \end{cases}$$

This ideal filter is non-causal and cannot be realized. The unit sample response h(n) is infinite in duration and must be truncated depending on the expected length M of the filter. This truncation is equivalent to the multiplication of the impulse response by a window function w(n).

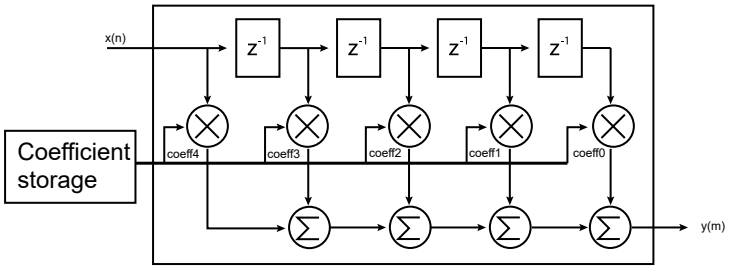
**Table 38-46. Window Function for a Filter Length M**

| Name of Window Function | Time Domain Sequence w(n) |
|---|---|
| Barlett | $1 - \frac{2 \times \left\lvert n - \frac{M-1}{2} \right\rvert}{M-1}$ |
| Blackman | $0.42 - 0.5 \times \cos\frac{2\pi n}{M-1} + 0.08 \times \cos\frac{4\pi n}{M-1}$ |

| .....continued | |
|---|---|
| **Name of Window Function** | **Time Domain Sequence w(n)** |
| Hamming | $0.54 - 0.46 \times \cos\frac{2\pi n}{M-1}$ |
| Hanning | $0.5 - 0.5 \times \cos\frac{2\pi n}{M-1}$ |

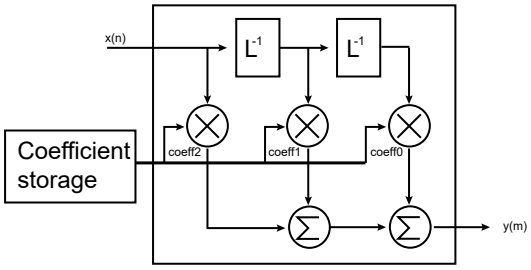
The horizontal resampler includes an 8-phase 5-tap filter equivalent to a 40-tap FIR described in the figure below.

**Figure 38-7. Horizontal Resampler Filter Architecture**




The vertical resampler includes an 8-phase 3-tap filter equivalent to a 24-tap FIR described in the figure below.

**Figure 38-8. Vertical Resampler Filter Architecture**




### 38.6.9.2 Horizontal Scaler

The XMEMSIZE field of the LCDC_HEOCFG4 register indicates the horizontal size minus one of the image in the system memory. The XSIZE field of the LCDC_HEOCFG3 register contains the horizontal size minus one of the window. The SCALEN bit of the LCDC_HEOCFG13 register is set to '1'. The scaling factor is programmed in the XFACTOR field of the LCDC_HEOCFG13 register. Use the following algorithm to find the XFACTOR value:

$$\mathrm{XFACTO}R_{1\mathrm{st}} = \mathrm{floor}\left(\frac{8 \times 256 \times \mathrm{XMEMSIZE} - 256 \times \mathrm{XPHIDEF}}{\mathrm{XSIZE}}\right)$$

$$\mathrm{XFACTO}R_{1\mathrm{st}} = \mathrm{XFACTO}R_{1\mathrm{st}} + 1$$

$$\mathrm{XMEMSIZE}_{\max} = \mathrm{floor}\left(\frac{\mathrm{XFACTO}R_{1\mathrm{st}} \times \mathrm{XSIZE} + 256 \times \mathrm{XPHIDEF}}{2048}\right)$$

$$\begin{cases} \mathrm{XFACTOR} = \mathrm{XFACTO}R_{1\mathrm{st}} - 1 \mathrm{when}(\mathrm{XMEMSIZE}_{\max} > \mathrm{XMEMSIZE}) \\ \mathrm{XFACTOR} = \mathrm{XFACTO}R_{1\mathrm{st}} \mathrm{otherwise} \end{cases}$$

#### 38.6.9.3 Vertical Scaler

The YMEMSIZE field of the LCDC_HEOCFG4 register indicates the vertical size minus one of the image in the system memory. The YSIZE field of the LCDC_HEOCFG3 register contains the vertical size minus one of the window. The SCALEN bit of the LCDC_HEOCFG13 register is set to one. The scaling factor is programmed in the YFACTOR field of the LCDC_HEOCFG13 register.

$$\mathrm{YFACTO}R_{1\mathrm{st}} = \mathrm{floor}\left(\frac{8 \times 256 \times \mathrm{YMEMSIZE} - 256 \times \mathrm{YPHIDEF}}{\mathrm{YSIZE}}\right)$$

$$\mathrm{YFACTO}R_{1\mathrm{st}} = \mathrm{YFACTO}R_{1\mathrm{st}} + 1$$

$$\mathrm{YMEMSIZE}_{\max} = \mathrm{floor}\left(\frac{\mathrm{YFACTO}R_{1\mathrm{st}} \times \mathrm{YSIZE} + 256 \times \mathrm{YPHIDEF}}{2048}\right)$$

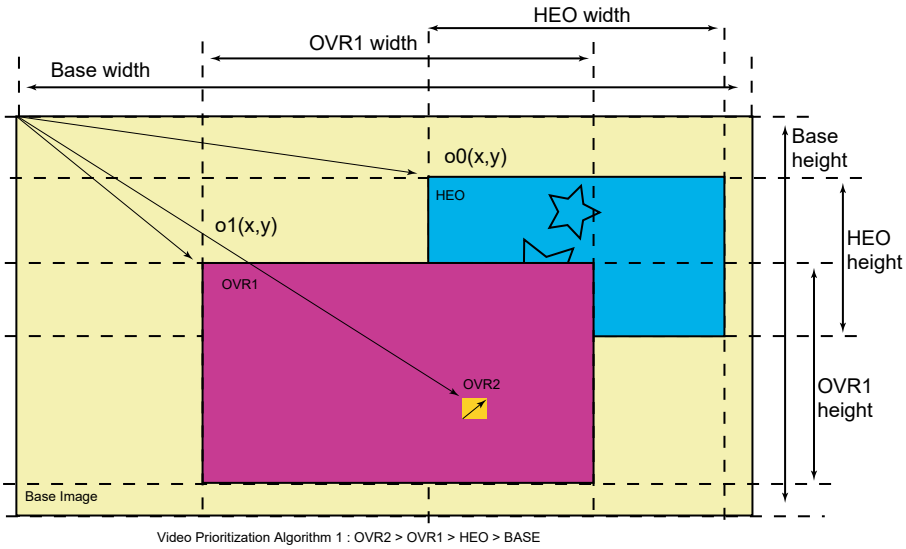
$$\begin{cases} \mathrm{YFACTOR} = \mathrm{YFACTO}R_{1\mathrm{st}} - 1 \mathrm{when}(\mathrm{YMEMSIZE}_{\max} > \mathrm{YMEMSIZE}) \\ \mathrm{YFACTOR} = \mathrm{YFACTO}R_{1\mathrm{st}} \mathrm{otherwise} \end{cases}$$

### 38.6.10 Color Combine Unit

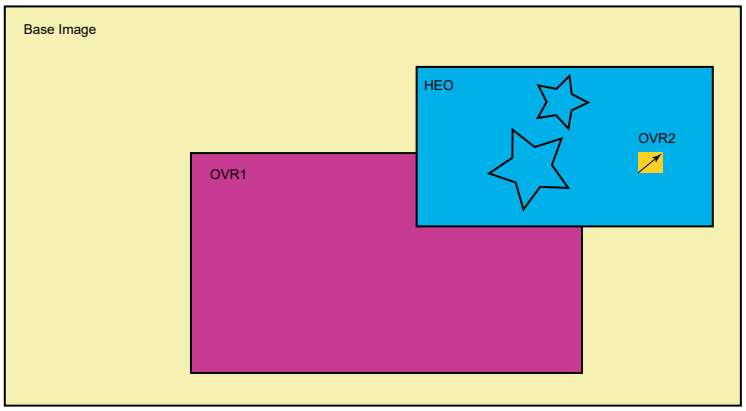
#### 38.6.10.1 Window Overlay

The LCD module provides hardware support for multiple “overlay plane” that can be used to display windows on top of the image without destroying the image located below. The overlay image can use any color depth. Using the overlay alleviates the need to re-render the occluded portion of the image. When pixels are combined together through the alpha blending unit, a new color is created. This new pixel is called an iterated pixel and is passed to the next blending stage. Then, this pixel may be combined again with another pixel. The VIDPRI bit located in the LCDC_HEOCFG12 register configures the video priority algorithm used to display the layers. When the VIDPRI bit is written to ‘0’, the OVR1 layer is located above the HEO layer. When the VIDPRI bit is written to ‘1’, OVR1 is located below the HEO layer.

**Figure 38-9. Overlay Example with Two Different Video Prioritization Algorithms**




Video Prioritization Algorithm 1 : OVR2 > OVR1 > HEO > BASE



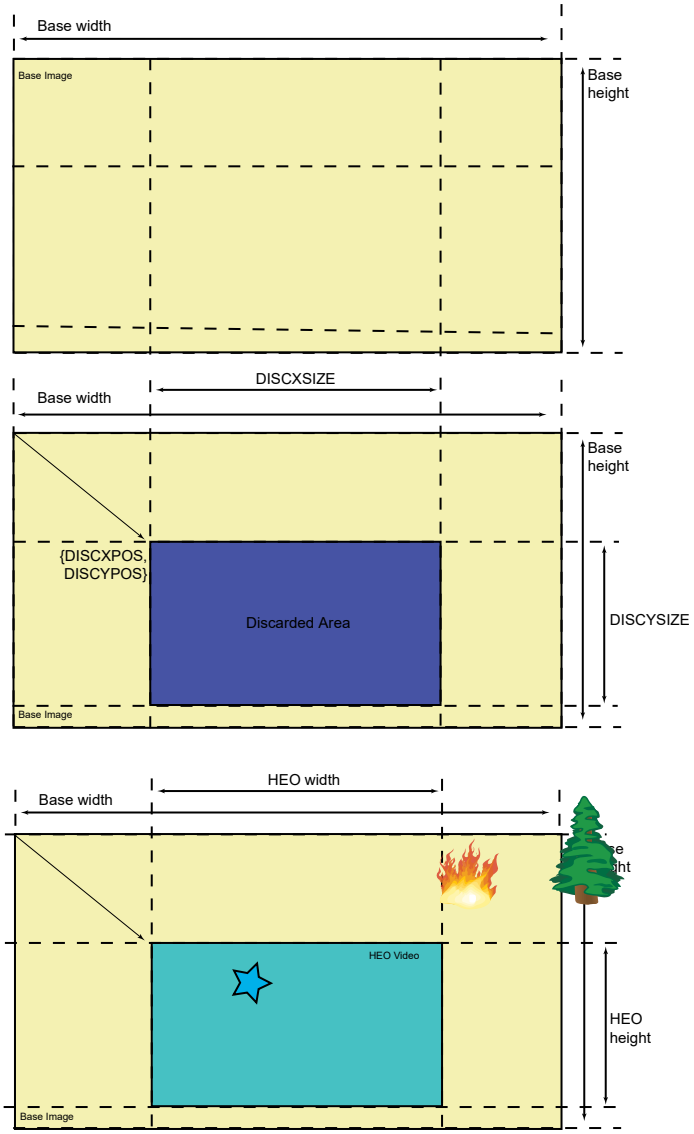

Video Prioritization Algorithm 2 : OVR2 > HEO > OVR1 > BASE

### 38.6.10.2 Base Layer with Window Overlay Optimization

When the base layer is combined with at least one active overlay, the whole base layer frame is retrieved from the memory though it is not visible. A set of registers is used to disable the Base DMA when this condition is met. These registers are the following:

- LCDC_BASECFG5:
  - field DISCXPOS (Discard Area Horizontal Position)
  - field DISCYPOS (Discard Area Vertical Position)
- LCDC_BASECFG6:
  - field DISCXSIZE (Discard Area Horizontal Size)
  - field DISCYSIZE (Discard Area Vertical Size)
- LCDC_BASECFG4: bit DISCEN (Discard Area Enable)

**Figure 38-10. Base Layer Discard Area**




### 38.6.10.3 Overlay Blending

The blending function requires two pixels (one iterated from the previous blending stage and one from the current overlay color) and a set of blending configuration parameters. These parameters define the color operation.

**Figure 38-11. Alpha Blender Function**

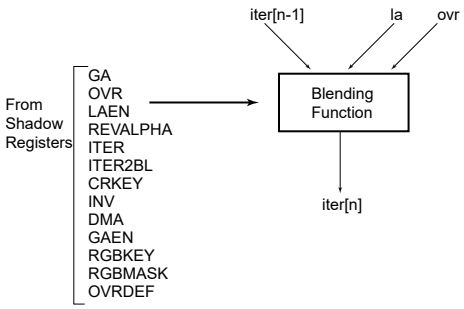
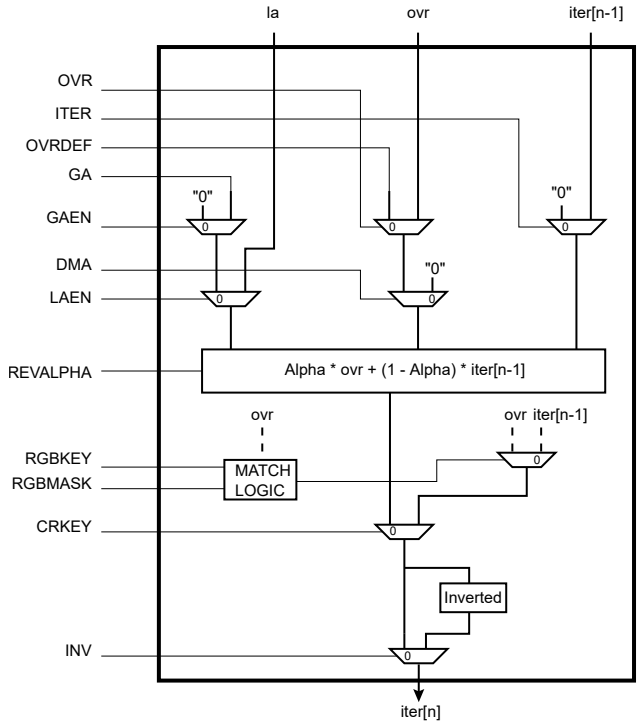
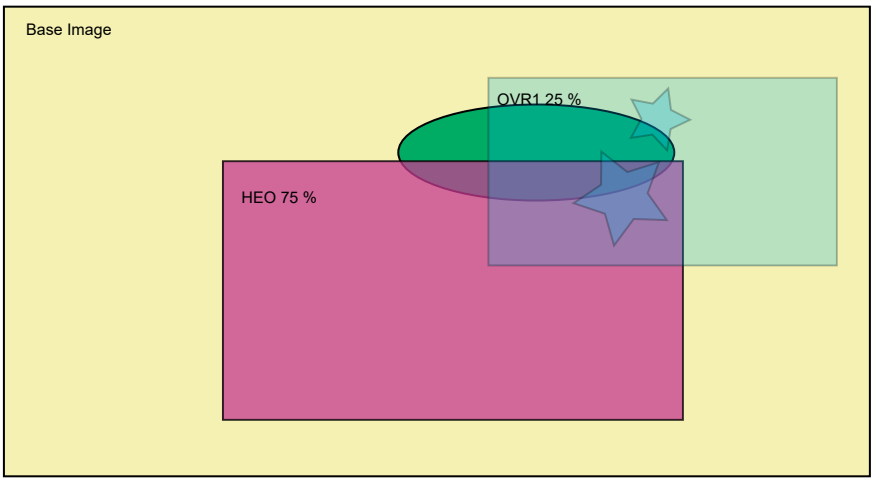


**Figure 38-12. Alpha Blender Database**




### 38.6.10.4 Window Blending

**Figure 38-13. 256-level Alpha Blending**




Video Prioritization Algorithm 1: OVR1 > HEO > BASE

### 38.6.10.5 Color Keying

Color keying involves a method of bit-block image transfer (Blit). This entails blitting one image onto another where not all the pixels are copied. Blitting usually involves two bitmaps: a source bitmap and a destination bitmap. A raster operation (ROP) is performed to define whether the iterated color or the overlay color is to be visible or not.

#### 38.6.10.5.1 Source Color Keying

If the masked overlay color matches the color key, the iterated color is selected and Source Color Keying is activated using the following configuration sequence:

1. Select the overlay to blit.
2. Write a ‘0’ to DSTKEY.
3. Activate Color Keying by writing a ‘1’ to CRKEY.
4. Configure the Color Key by writing RKEY, GKEY and BKEY fields.
5. Configure the Color Mask by writing RKEY, GKEY and BKEY fields.

When the field RMASK, GMASK, or BMASK is configured to ‘0’, the comparison is disabled and the raster operation is activated.

#### 38.6.10.5.2 Destination Color Keying

If the iterated masked color matches the color key then the overlay color is selected, Destination Color Keying is activated using the following configuration sequence:

1. Select the overlay to blit.
2. Write a ‘1’ to DSTKEY.
3. Activate Color Keying by writing a ‘1’ to CRKEY bit
4. Configure the Color Key by writing RKEY, GKEY and BKEY fields.
5. Configure the Color Mask by writing RKEY, GKEY and BKEY fields.

When the field RMASK, GMASK, or BMASK is configured to ‘0’, the comparison is disabled and the raster operation is activated.

### 38.6.11 LCDC PWM Controller

This block generates the LCD contrast control signal (LCDPWM) to make possible the control of the display's contrast by software. This is an 8-bit PWM (Pulse Width Modulation) signal that can be converted to an analog voltage with a simple passive filter.

The PWM module has a free-running counter whose value is compared against a compare register (PWMCVAL field of the LCDC_LCDCFG6 register). If the value in the counter is less than that in the register, the output brings the value of the signal polarity (PWMPOL) bit in the PWM control register: LCDC_LCDCFG6. Otherwise, the opposite value is output. Thus, a periodic waveform with a pulse width proportional to the value in the compare register is generated.

Due to the comparison mechanism, the output pulse has a width between zero and 255 PWM counter cycles. Thus by adding a simple passive filter outside the chip, an analog voltage between 0 and (255/256) × $V_{DD}$ can be obtained (for the positive polarity case, or between (1/256) × $V_{DD}$ and $V_{DD}$ for the negative polarity case). Other voltage values can be obtained by adding active external circuitry.

For PWM mode, the counter frequency can be adjusted to four different values using the PWMPS field of the LCDC_LCDCFG6 register.

The PWM module can be fed with the slow clock or the system clock, depending on the CLKPWMSEL bit of the LCDC_CFG0 register.

LCD display panels have different backlight specifications in terms of minimum/maximum values for PWM frequency. if the LCDC PWM frequency range does not match the LCD display panel, it is possible to use the product standalone PWM controller to drive the backlight.

### 38.6.12 Post Processing Controller

The output stream of pixels can be either displayed on the screen or written to the memory using the Post Processing Controller (PPC). When the PPC is used, the screen display is disabled, but synchronization signals remain active (if enabled). The stream of pixel can be written in RGB mode or encoded in YCbCr 422 mode. A programmable color space conversion stage is available.

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} \text{CSCYR} & \text{CSCYG} & \text{CSCYB} \\ \text{CSCUR} & \text{CSCUG} & \text{CSCUB} \\ \text{CSCVR} & \text{CSCVG} & \text{CSCUB} \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} \text{Yoff} \\ \text{Uoff} \\ \text{Voff} \end{bmatrix}$$

### 38.6.13 LCD Overall Performance

#### 38.6.13.1 Color Lookup Table (CLUT)

**Table 38-47. CLUT Pixel Performance**

| CLUT Mode | Pixels/Cycle | Rotation | Scaling |
|---|---|---|---|
| 1 bpp | 64 | Not supported | Supported |
| 2 bpp | 32 | Not supported | Supported |
| 3 bpp | 16 | Not supported | Supported |
| 4 bpp | 8 | Not supported | Supported |

#### 38.6.13.2 RGB Mode Fetch Performance

**Table 38-48. RGB Mode Performance**

| RGB Mode | Pixels/Cycle Memory Burst Mode | Rotation Peak Random Memory Access (pixels/cycle) | | Scaling Burst Mode or Rotation Optimization Available |
|---|---|---|---|---|
| | | Rotation Optimization (see table footnote) | Normal Mode | |
| 12 bpp | 4 | 1 | 0.2 | Supported |
| 16 bpp | 4 | 1 | 0.2 | Supported |
| 18 bpp | 2 | 1 | 0.2 | Supported |
| 18 bpp RGB PACKED | 2.666 | Not supported | 0.2 | Supported |
| 19 bpp | 2 | 1 | 0.2 | Supported |
| 19 bpp PACKED | 2.666 | Not Supported | 0.2 | Supported |
| 24 bpp | 2 | 1 | 0.2 | Supported |
| 24 bpp PACKED | 2.666 | Not Supported | 0.2 | Supported |
| 25 bpp | 2 | 1 | 0.2 | Supported |
| 32 bpp | 2 | 1 | 0.2 | Supported |

**Note:** Rotation optimization = AHB lock asserted on consecutive single access.

#### 38.6.13.3 YUV Mode Fetch Performance

**Table 38-49. Single Stream for 0 Wait State Memory**

| YUV Mode | Pixels/Cycle Memory Burst Mode | Rotation Peak Random Memory Access (pixels/cycle) | | Scaling Burst Mode or Rotation Optimization Is Available |
|---|---|---|---|---|
| | | Rotation Optimization (see Note 1) | Normal Mode | |
| 32 bpp AYUV | 2 | 1 | 0.2 | Supported |
| 16 bpp 422 | 4 | Not Supported | Not Supported | Supported |

**Note:**

1. Rotation optimization = AHB lock asserted on consecutive single access.

**Table 38-50. Multiple Stream for 0 Wait State Memory**

| YUV Mode | Comp/Cycle Memory Burst Mode | Rotation Peak Random Memory Access (pixels/cycle) | | Scaling Burst Mode or Rotation Optimization Is Available |
|---|---|---|---|---|
| | | Rotation Optimization | Normal Mode | |
| 16 bpp 422 semiplanar | 8 Y, 4 UV | 1 Y, 1 UV (2 streams) | 0.2 Y 0.2 UV (2 streams) | Supported |
| 16 bpp 422 planar | 8 Y, 8 U, 8 V | 1 Y, 1 U, 1 V (3 streams) | 0.2 Y, 0.2 U, 0.2 V (3 streams) | Supported |
| 12 bpp 4:2:0 semiplanar | 8 Y, 4 UV | 1 Y, 1 UV (2 streams) | 0.2 Y 0.2 UV (2 streams) | Supported |
| 12 bpp 4:2:0 planar | 8 Y, 8 U, 8 V | 1 Y, 1 U, 1 V (3 streams) | 0.2 Y, 0.2 U, 0.2 V (3 streams) | Supported |

**Note:** In order to provide more bandwidth when multiple streams are used to transfer Y, UV, U or V components, two AHB interfaces are recommended or multiple AXI IDs are required.

**Table 38-51. YUV Planar Overall Performance 1 AHB Interface for 0 Wait State Memory**

| YUV Mode | Pix/Cycle Memory Burst Mode | Rotation Peak Random Memory Access (pixels/cycle) | | Scaling Burst Mode or Rotation Optimization Is Available |
|---|---|---|---|---|
| | | Rotation Optimization | Normal Mode | |
| 16 bpp 422 semiplanar | 4 | 0.66 | 0.132 | Supported |
| 16 bpp 422 planar | 4 | 0.5 | 0.1 | Supported |
| 12 bpp 4:2:0 semiplanar | 5.32 | 0.8 | 0.16 | Supported |
| 12 bpp 4:2:0 planar | 5.32 | 0.66 | 0.132 | Supported |

**Note:** To provide more bandwidth when multiple streams are used to transfer Y, UV, U or V components, two AHB interfaces are recommended or multiple AXI IDs are required.

### 38.6.14 Input FIFO

The LCD module includes one input FIFO per overlay. These input FIFOs are used to buffer the AHB burst and serialize the stream of pixels.

### 38.6.15 Output FIFO

The LCD module includes one output FIFO that stores the blended pixel.

### 38.6.16 Output Timing Generation

#### 38.6.16.1 Active Display Timing Mode

**Figure 38-14. Active Display Timing**

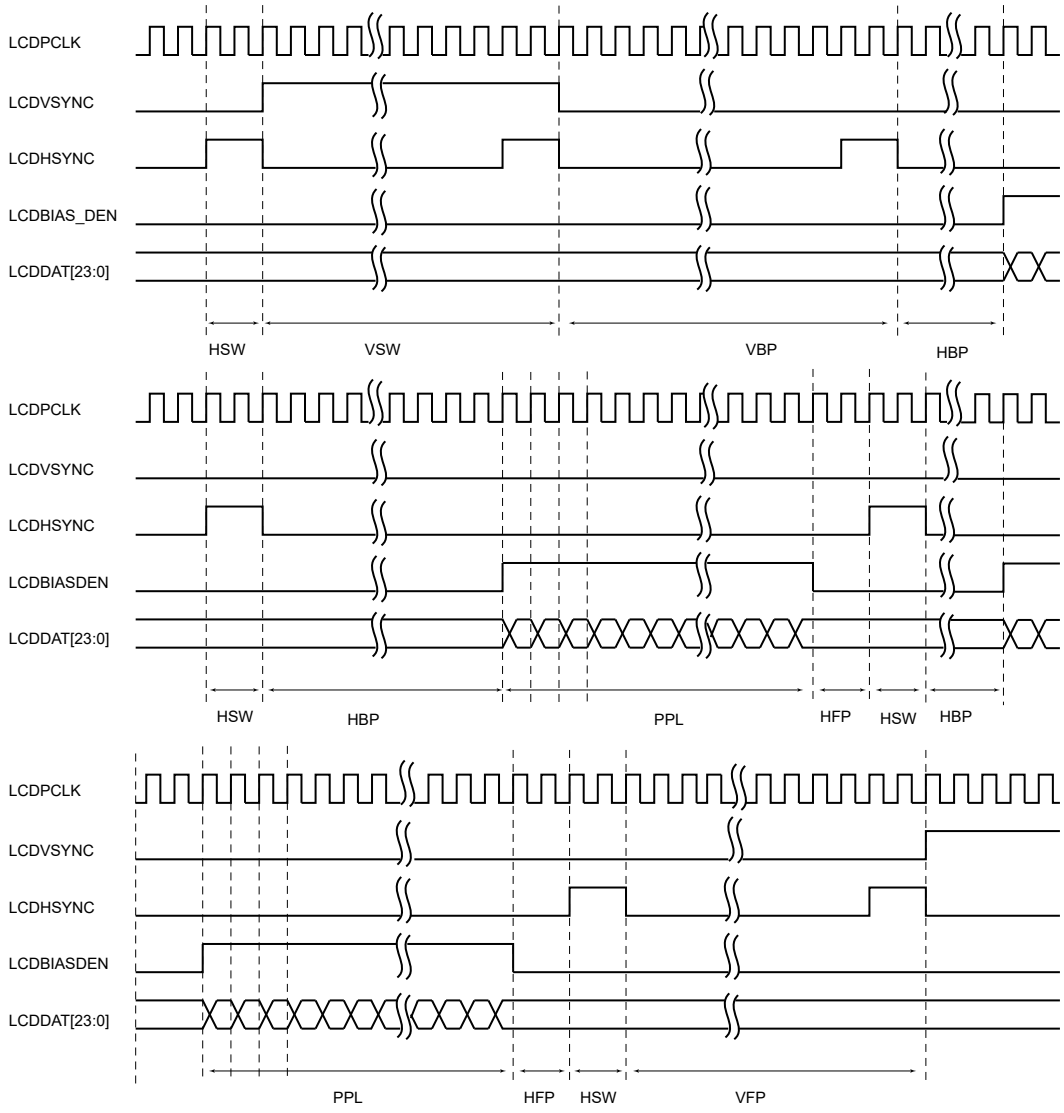


**Figure 38-15. Vertical Synchronization Timing (part 1)**

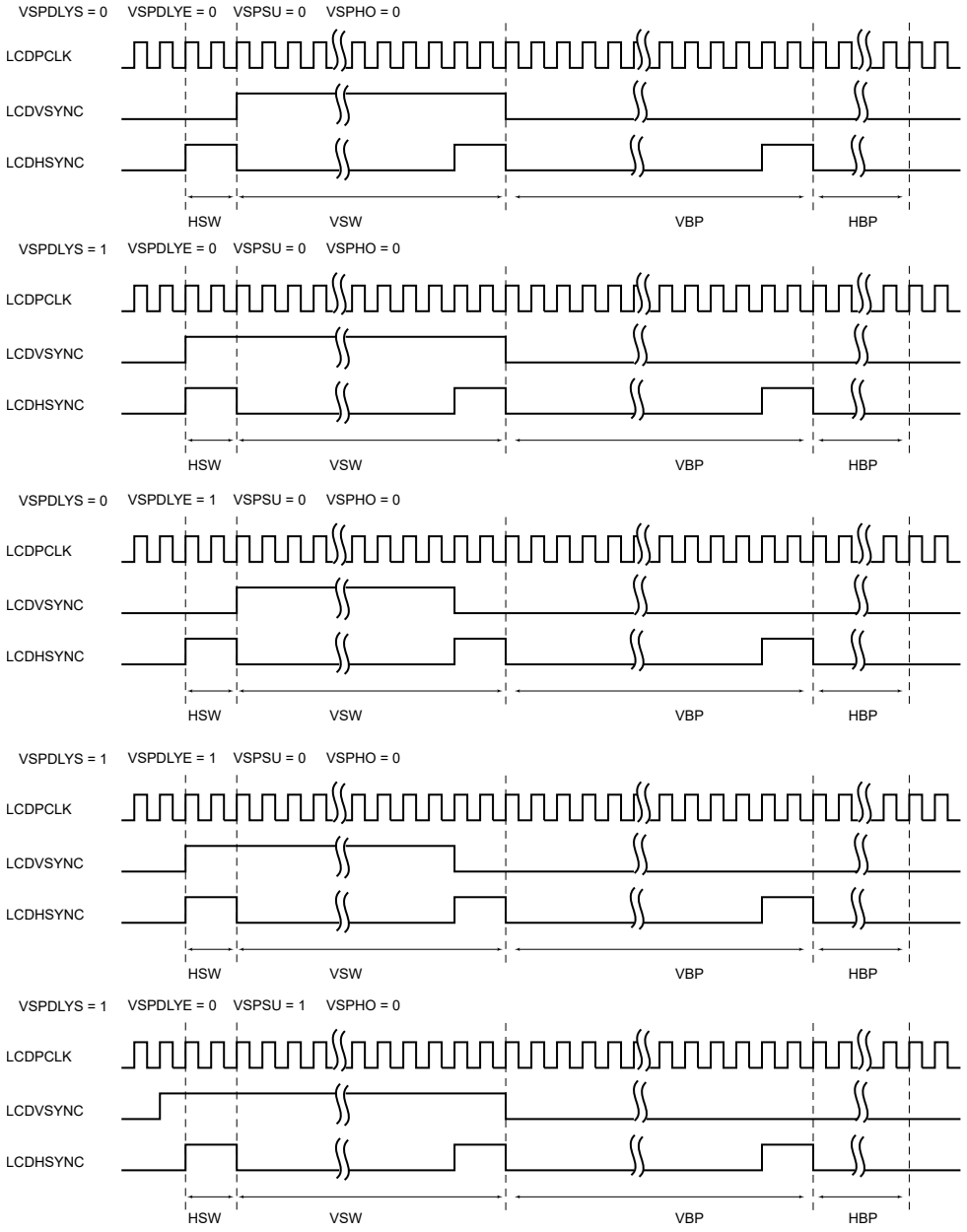


**Figure 38-16. Vertical Synchronization Timing (part 2)**

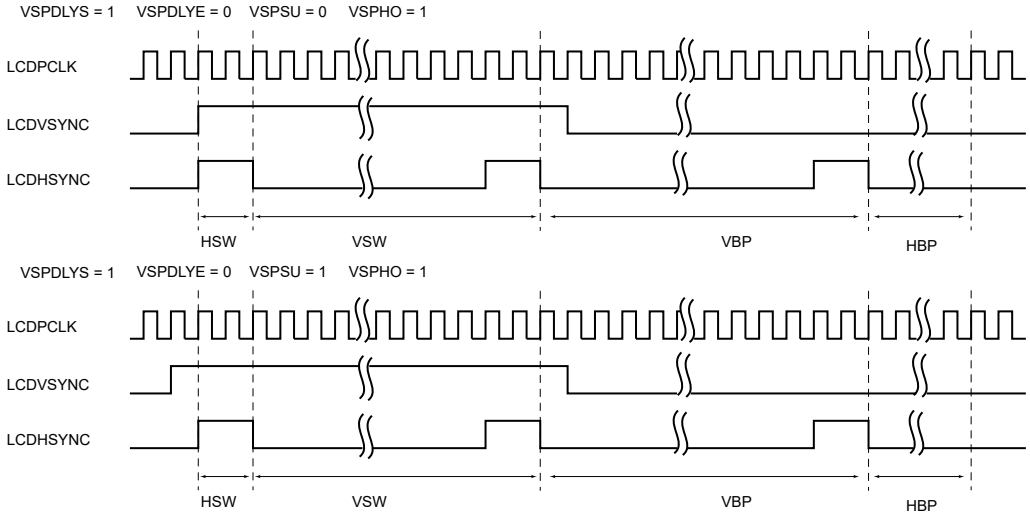
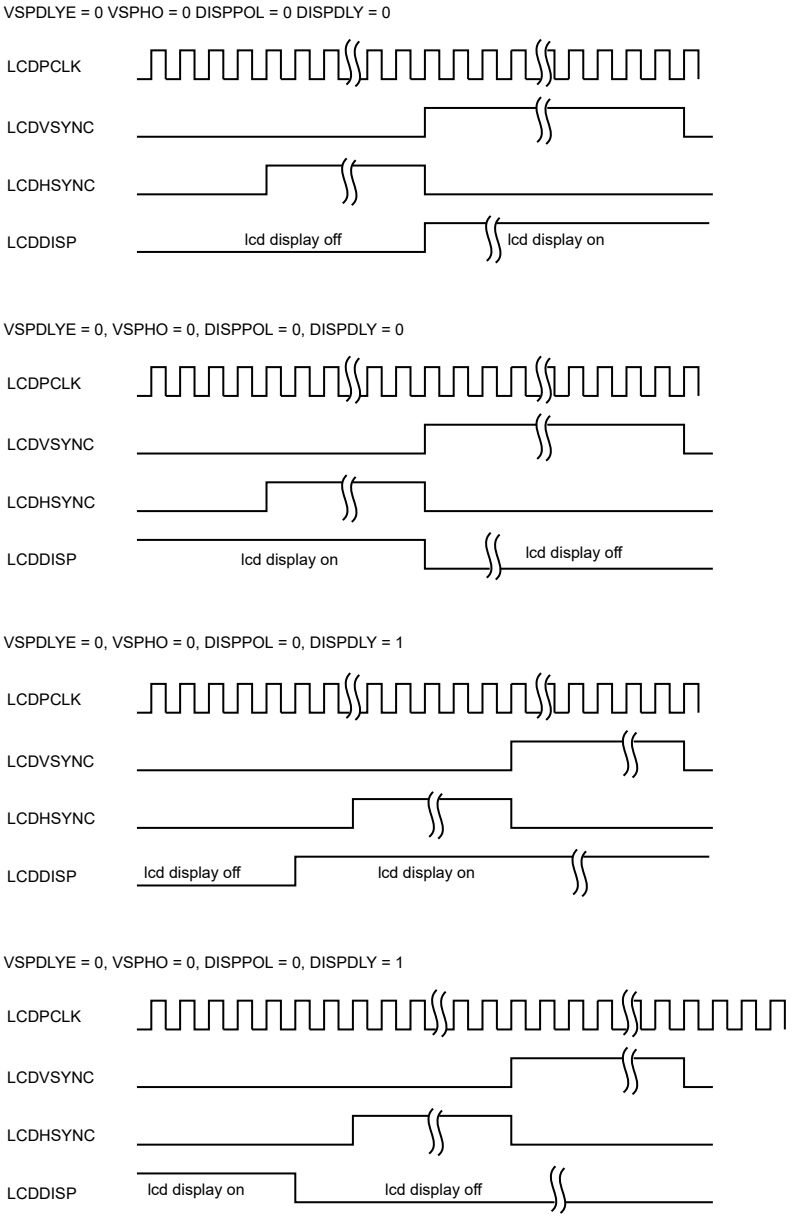


**Figure 38-17. DISP Signal Timing Diagram**




### 38.6.17 Output Format

#### 38.6.17.1 Active Mode Output Pin Assignment

**Table 38-52. Active Mode Output with 24-bit Bus Interface Configuration**

| Pin ID | TFT 24 bits | TFT 18 bits | TFT 16 bits | TFT 12 bits |
|---|---|---|---|---|
| LCDDAT[23] | R[7] | R[5] | R[4] | R[3] |
| LCDDAT[22] | R[6] | R[4] | R[3] | R[2] |
| LCDDAT[21] | R[5] | R[3] | R[2] | R[1] |
| LCDDAT[20] | R[4] | R[2] | R[1] | R[0] |

.....continued

| Pin ID | TFT 24 bits | TFT 18 bits | TFT 16 bits | TFT 12 bits |
|---|---|---|---|---|
| LCDDAT[19] | R[3] | R[1] | R[0] | - |
| LCDDAT[18] | R[2] | R[0] | - | - |
| LCDDAT[17] | R[1] | - | - | - |
| LCDDAT[16] | R[0] | - | - | - |
| LCDDAT[15] | G[7] | G[5] | G[5] | G[3] |
| LCDDAT[14] | G[6] | G[4] | G[4] | G[2] |
| LCDDAT[13] | G[5] | G[3] | G[3] | G[1] |
| LCDDAT[12] | G[4] | G[2] | G[2] | G[0] |
| LCDDAT[11] | G[3] | G[1] | G[1] | - |
| LCDDAT[10] | G[2] | G[0] | G[0] | - |
| LCDDAT[9] | G[1] | - | - | - |
| LCDDAT[8] | G[0] | - | - | - |
| LCDDAT[7] | B[7] | B[5] | B[4] | B[3] |
| LCDDAT[6] | B[6] | B[4] | B[3] | B[2] |
| LCDDAT[5] | B[5] | B[3] | B[2] | B[1] |
| LCDDAT[4] | B[4] | B[2] | B[1] | B[0] |
| LCDDAT[3] | B[3] | B[1] | B[0] | - |
| LCDDAT[2] | B[2] | B[0] | - | - |
| LCDDAT[1] | B[1] | - | - | - |
| LCDDAT[0] | B[0] | - | - | - |

## 38.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | LCDC_LCDCFG0 | 7:0 | | | | | CLKPWMSEL | CLKSEL | | CLKPOL |
| | | 15:8 | | | CGDISPP | | CGDISHEO | CGDISOVR2 | CGDISOVR1 | CGDISBASE |
| | | 23:16 | CLKDIV[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | LCDC_LCDCFG1 | 7:0 | HSPW[7:0] | | | | | | | |
| | | 15:8 | | | | | | | HSPW[9:8] | |
| | | 23:16 | VSPW[7:0] | | | | | | | |
| | | 31:24 | | | | | | | VSPW[9:8] | |
| 0x08 | LCDC_LCDCFG2 | 7:0 | VFPW[7:0] | | | | | | | |
| | | 15:8 | | | | | | | VFPW[9:8] | |
| | | 23:16 | VBPW[7:0] | | | | | | | |
| | | 31:24 | | | | | | | VBPW[9:8] | |
| 0x0C | LCDC_LCDCFG3 | 7:0 | HFPW[7:0] | | | | | | | |
| | | 15:8 | | | | | | | HFPW[9:8] | |
| | | 23:16 | HBPW[7:0] | | | | | | | |
| | | 31:24 | | | | | | | HBPW[9:8] | |
| 0x10 | LCDC_LCDCFG4 | 7:0 | PPL[7:0] | | | | | | | |
| | | 15:8 | | | | | | PPL[10:8] | | |
| | | 23:16 | RPF[7:0] | | | | | | | |
| | | 31:24 | | | | | | RPF[10:8] | | |
| 0x14 | LCDC_LCDCFG5 | 7:0 | DISPDLY | DITHER | | DISPPOL | VSPDLYE | VSPDLYS | VSPOL | HSPOL |
| | | 15:8 | | | VSPHO | VSPSU | | PP | MODE[1:0] | |
| | | 23:16 | GUARDTIME[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | LCDC_LCDCFG6 | 7:0 | | | | PWMPOL | | PWMPS[2:0] | | |
| | | 15:8 | PWMCVAL[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C ... 0x1F | Reserved | | | | | | | | | |
| 0x20 | LCDC_LCDEN | 7:0 | | | | | PWMEN | DISPEN | SYNCEN | CLKEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 | LCDC_LCDDIS | 7:0 | | | | | PWMDIS | DISPDIS | SYNCDIS | CLKDIS |
| | | 15:8 | | | | | PWMRST | DISPRST | SYNCRST | CLKRST |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x28 | LCDC_LCDSR | 7:0 | | | | SIPSTS | PWMSTS | DISPSTS | LCDSTS | CLKSTS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x2C | LCDC_LCDIER | 7:0 | | | | FIFOERRIE | | DISPIE | DISIE | SOFIE |
| | | 15:8 | | | PPIE | | HEOIE | OVR2IE | OVR1IE | BASEIE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x30 | LCDC_LCDIDR | 7:0 | | | | FIFOERRID | | DISPID | DISID | SOFID |
| | | 15:8 | | | PPID | | HEOID | OVR2ID | OVR1ID | BASEID |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x34 | LCDC_LCDIMR | 7:0 | | | | FIFOERRIM | | DISPIM | DISIM | SOFIM |
| | | 15:8 | | | PPIM | | HEOIM | OVR2IM | OVR1IM | BASEIM |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x38 | LCDC_LCDISR | 7:0 | | | | FIFOERR | | DISP | DIS | SOF |
| | | 15:8 | | | PP | | HEO | OVR2 | OVR1 | BASE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x3C | LCDC_ATTR | 7:0 | | | PP | | HEO | OVR2 | OVR1 | BASE |
| | | 15:8 | | | PPA2Q | | HEOA2Q | OVR2A2Q | OVR1A2Q | BASEA2Q |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x40 | LCDC_BASECHER | 7:0 | | | | | | A2QEN | UPDATEEN | CHEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x40 | LCDC_BASECHDR | 7:0 | | | | | | | | CHDIS |
| | | 15:8 | | | | | | | | CHRST |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x44 ... 0x47 | Reserved | | | | | | | | | |
| 0x48 | LCDC_BASECHSR | 7:0 | | | | | | A2QSR | UPDATESR | CHSR |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x4C | LCDC_BASEIER | 7:0 | | OVR | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x50 | LCDC_BASEIDR | 7:0 | | OVR | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x54 | LCDC_BASEIMR | 7:0 | | OVR | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x58 | LCDC_BASEISR | 7:0 | | OVR | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x5C | LCDC_BASEHEAD | 7:0 | HEAD[5:0] | | | | | | | |
| | | 15:8 | HEAD[13:6] | | | | | | | |
| | | 23:16 | HEAD[21:14] | | | | | | | |
| | | 31:24 | HEAD[29:22] | | | | | | | |
| 0x60 | LCDC_BASEADDR | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0x64 | LCDC_BASECTRL | 7:0 | | | DONEIEN | ADDIEN | DSCRIEN | DMAIEN | LFETCH | DFETCH |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x68 | LCDC_BASENEXT | 7:0 | NEXT[7:0] | | | | | | | |
| | | 15:8 | NEXT[15:8] | | | | | | | |
| | | 23:16 | NEXT[23:16] | | | | | | | |
| | | 31:24 | NEXT[31:24] | | | | | | | |
| 0x6C | LCDC_BASECFG0 | 7:0 | | | BLEN[1:0] | | | | | SIF |
| | | 15:8 | | | | | | | | DLBO |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x70 | LCDC_BASECFG1 | 7:0 | RGBMODE[3:0] | | | | | | | CLUTEN |
| | | 15:8 | | | | | | | CLUTMODE[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x74 | LCDC_BASECFG2 | 7:0 | XSTRIDE[7:0] | | | | | | | |
| | | 15:8 | XSTRIDE[15:8] | | | | | | | |
| | | 23:16 | XSTRIDE[23:16] | | | | | | | |
| | | 31:24 | XSTRIDE[31:24] | | | | | | | |
| 0x78 | LCDC_BASECFG3 | 7:0 | BDEF[7:0] | | | | | | | |
| | | 15:8 | GDEF[7:0] | | | | | | | |
| | | 23:16 | RDEF[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x7C | LCDC_BASECFG4 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | DISCEN | | REP | DMA |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x80 | LCDC_BASECFG5 | 7:0 | DISCXPOS[7:0] | | | | | | | |
| | | 15:8 | | | | | | DISCXPOS[10:8] | | |
| | | 23:16 | DISCYPOS[7:0] | | | | | | | |
| | | 31:24 | | | | | | DISCYPOS[10:8] | | |
| 0x84 | LCDC_BASECFG6 | 7:0 | DISCXSIZE[7:0] | | | | | | | |
| | | 15:8 | | | | | | DISCXSIZE[10:8] | | |
| | | 23:16 | DISCYSIZE[7:0] | | | | | | | |
| | | 31:24 | | | | | | DISCYSIZE[10:8] | | |
| 0x88 ... 0x013F | Reserved | | | | | | | | | |
| 0x0140 | LCDC_OVR1CHER | 7:0 | | | | | | A2QEN | UPDATEEN | CHEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0144 | LCDC_OVR1CHDR | 7:0 | | | | | | | | CHDIS |
| | | 15:8 | | | | | | | | CHRST |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0148 | LCDC_OVR1CHSR | 7:0 | | | | | | A2QSR | UPDATESR | CHSR |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x014C | LCDC_OVR1IER | 7:0 | | OVR | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0150 | LCDC_OVR1IDR | 7:0 | | OVR | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0154 | LCDC_OVR1IMR | 7:0 | | OVR | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0158 | LCDC_OVR1ISR | 7:0 | | OVR | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x015C | LCDC_OVR1HEAD | 7:0 | HEAD[5:0] | | | | | | | |
| | | 15:8 | HEAD[13:6] | | | | | | | |
| | | 23:16 | HEAD[21:14] | | | | | | | |
| | | 31:24 | HEAD[29:22] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0160 | LCDC_OVR1ADDR | 7:0 | ADDR[7:0] | | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | | |
| 0x0164 | LCDC_OVR1CTRL | 7:0 | | | DONEIEN | ADDIEN | DSCRIEN | DMAIEN | LFETCH | DFETCH | |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x0168 | LCDC_OVR1NEXT | 7:0 | NEXT[7:0] | | | | | | | | |
| | | 15:8 | NEXT[15:8] | | | | | | | | |
| | | 23:16 | NEXT[23:16] | | | | | | | | |
| | | 31:24 | NEXT[31:24] | | | | | | | | |
| 0x016C ... 0x016F | Reserved | | | | | | | | | | |
| 0x0170 | LCDC_OVR1CFG0 | 7:0 | | | BLEN[1:0] | | | | | SIF | |
| | | 15:8 | | | LOCKDIS | ROTDIS | | | | DLBO | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x0170 | LCDC_OVR1CFG1 | 7:0 | RGBMODE[3:0] | | | | | | | CLUTEN | |
| | | 15:8 | | | | | | | CLUTMODE[1:0] | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x0174 | LCDC_OVR1CFG2 | 7:0 | XPOS[7:0] | | | | | | | | |
| | | 15:8 | | | | | | XPOS[10:8] | | | |
| | | 23:16 | YPOS[7:0] | | | | | | | | |
| | | 31:24 | | | | | | YPOS[10:8] | | | |
| 0x0178 | LCDC_OVR1CFG3 | 7:0 | XSIZE[7:0] | | | | | | | | |
| | | 15:8 | | | | | | XSIZE[10:8] | | | |
| | | 23:16 | YSIZE[7:0] | | | | | | | | |
| | | 31:24 | | | | | | YSIZE[10:8] | | | |
| 0x017C | LCDC_OVR1CFG4 | 7:0 | XSTRIDE[7:0] | | | | | | | | |
| | | 15:8 | XSTRIDE[15:8] | | | | | | | | |
| | | 23:16 | XSTRIDE[23:16] | | | | | | | | |
| | | 31:24 | XSTRIDE[31:24] | | | | | | | | |
| 0x0180 | LCDC_OVR1CFG5 | 7:0 | PSTRIDE[7:0] | | | | | | | | |
| | | 15:8 | PSTRIDE[15:8] | | | | | | | | |
| | | 23:16 | PSTRIDE[23:16] | | | | | | | | |
| | | 31:24 | PSTRIDE[31:24] | | | | | | | | |
| 0x0184 | LCDC_OVR1CFG6 | 7:0 | BDEF[7:0] | | | | | | | | |
| | | 15:8 | GDEF[7:0] | | | | | | | | |
| | | 23:16 | RDEF[7:0] | | | | | | | | |
| | | 31:24 | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0188 | LCDC_OVR1CFG7 | 7:0 | BKEY[7:0] | | | | | | | |
| | | 15:8 | GKEY[7:0] | | | | | | | |
| | | 23:16 | RKEY[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x018C | LCDC_OVR1CFG8 | 7:0 | BMASK[7:0] | | | | | | | |
| | | 15:8 | GMASK[7:0] | | | | | | | |
| | | 23:16 | RMASK[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0190 | LCDC_OVR1CFG9 | 7:0 | OVR | LAEN | GAEN | REVALPHA | ITER | ITER2BL | INV | CRKEY |
| | | 15:8 | | | | | | DSTKEY | REP | DMA |
| | | 23:16 | GA[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0194 ... 0x023F | Reserved | | | | | | | | | |
| 0x0240 | LCDC_OVR2CHER | 7:0 | | | | | | A2QEN | UPDATEEN | CHEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0244 | LCDC_OVR2CHDR | 7:0 | | | | | | | | CHDIS |
| | | 15:8 | | | | | | | | CHRST |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0248 | LCDC_OVR2CHSR | 7:0 | | | | | | A2QSR | UPDATESR | CHSR |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x024C | LCDC_OVR2IER | 7:0 | | OVR | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0250 | LCDC_OVR2IDR | 7:0 | | OVR | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0254 | LCDC_OVR2IMR | 7:0 | | OVR | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0258 | LCDC_OVR2ISR | 7:0 | | OVR | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x025C | LCDC_OVR2HEAD | 7:0 | HEAD[5:0] | | | | | | | |
| | | 15:8 | HEAD[13:6] | | | | | | | |
| | | 23:16 | HEAD[21:14] | | | | | | | |
| | | 31:24 | HEAD[29:22] | | | | | | | |
| 0x0260 | LCDC_OVR2ADDR | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0x0264 | LCDC_OVR2CTRL | 7:0 | | | DONEIEN | ADDIEN | DSCRIEN | DMAIEN | LFETCH | DFETCH |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0268 | LCDC_OVR2NEXT | 7:0 | NEXT[7:0] | | | | | | | |
| | | 15:8 | NEXT[15:8] | | | | | | | |
| | | 23:16 | NEXT[23:16] | | | | | | | |
| | | 31:24 | NEXT[31:24] | | | | | | | |
| 0x026C | LCDC_OVR2CFG0 | 7:0 | | | BLEN[1:0] | | | | | |
| | | 15:8 | | | LOCKDIS | ROTDIS | | | | DLBO |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0270 | LCDC_OVR2CFG1 | 7:0 | RGBMODE[3:0] | | | | | | | CLUTEN |
| | | 15:8 | | | | | | | CLUTMODE[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0274 | LCDC_OVR2CFG2 | 7:0 | XPOS[7:0] | | | | | | | |
| | | 15:8 | | | | | | XPOS[10:8] | | |
| | | 23:16 | YPOS[7:0] | | | | | | | |
| | | 31:24 | | | | | | YPOS[10:8] | | |
| 0x0278 | LCDC_OVR2CFG3 | 7:0 | XSIZE[7:0] | | | | | | | |
| | | 15:8 | | | | | | XSIZE[10:8] | | |
| | | 23:16 | YSIZE[7:0] | | | | | | | |
| | | 31:24 | | | | | | YSIZE[10:8] | | |
| 0x027C | LCDC_OVR2CFG4 | 7:0 | XSTRIDE[7:0] | | | | | | | |
| | | 15:8 | XSTRIDE[15:8] | | | | | | | |
| | | 23:16 | XSTRIDE[23:16] | | | | | | | |
| | | 31:24 | XSTRIDE[31:24] | | | | | | | |
| 0x0280 | LCDC_OVR2CFG5 | 7:0 | PSTRIDE[7:0] | | | | | | | |
| | | 15:8 | PSTRIDE[15:8] | | | | | | | |
| | | 23:16 | PSTRIDE[23:16] | | | | | | | |
| | | 31:24 | PSTRIDE[31:24] | | | | | | | |
| 0x0284 | LCDC_OVR2CFG6 | 7:0 | BDEF[7:0] | | | | | | | |
| | | 15:8 | GDEF[7:0] | | | | | | | |
| | | 23:16 | RDEF[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0288 | LCDC_OVR2CFG7 | 7:0 | BKEY[7:0] | | | | | | | |
| | | 15:8 | GKEY[7:0] | | | | | | | |
| | | 23:16 | RKEY[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x028C | LCDC_OVR2CFG8 | 7:0 | BMASK[7:0] | | | | | | | |
| | | 15:8 | GMASK[7:0] | | | | | | | |
| | | 23:16 | RMASK[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0290 | LCDC_OVR2CFG9 | 7:0 | OVR | LAEN | GAEN | REVALPHA | ITER | ITER2BL | INV | CRKEY |
| | | 15:8 | | | | | | DSTKEY | REP | DMA |
| | | 23:16 | GA[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0294 ... 0x033F | Reserved | | | | | | | | | |
| 0x0340 | LCDC_HEOCHER | 7:0 | | | | | | A2QEN | UPDATEEN | CHEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0344 | LCDC_HEOCHDR | 7:0 | | | | | | | | CHDIS |
| | | 15:8 | | | | | | | | CHRST |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0348 | LCDC_HEOCHSR | 7:0 | | | | | | A2QSR | UPDATESR | CHSR |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x034C | LCDC_HEOIER | 7:0 | | OVR | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | UOVR | UDONE | UADD | UDSCR | UDMA | | |
| | | 23:16 | | VOVR | VDONE | VADD | VDSCR | VDMA | | |
| | | 31:24 | | | | | | | | |
| 0x0350 | LCDC_HEOIDR | 7:0 | | OVR | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | UOVR | UDONE | UADD | UDSCR | UDMA | | |
| | | 23:16 | | VOVR | VDONE | VADD | VDSCR | VDMA | | |
| | | 31:24 | | | | | | | | |
| 0x0354 | LCDC_HEOIMR | 7:0 | | OVR | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | UOVR | UDONE | UADD | UDSCR | UDMA | | |
| | | 23:16 | | VOVR | VDONE | VADD | VDSCR | VDMA | | |
| | | 31:24 | | | | | | | | |
| 0x0358 | LCDC_HEOISR | 7:0 | | OVR | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | UOVR | UDONE | UADD | UDSCR | UDMA | | |
| | | 23:16 | | VOVR | VDONE | VADD | VDSCR | VDMA | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x035C | LCDC_HEOHEAD | 7:0 | HEAD[5:0] | | | | | | | |
| | | 15:8 | HEAD[13:6] | | | | | | | |
| | | 23:16 | HEAD[21:14] | | | | | | | |
| | | 31:24 | HEAD[29:22] | | | | | | | |
| 0x0360 | LCDC_HEOADDR | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0x0364 | LCDC_HEOCTRL | 7:0 | | | DONEIEN | ADDIEN | DSCRIEN | DMAIEN | LFETCH | DFETCH |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0368 | LCDC_HEONEXT | 7:0 | NEXT[7:0] | | | | | | | |
| | | 15:8 | NEXT[15:8] | | | | | | | |
| | | 23:16 | NEXT[23:16] | | | | | | | |
| | | 31:24 | NEXT[31:24] | | | | | | | |
| 0x036C | LCDC_HEOUHEAD | 7:0 | UHEAD[7:0] | | | | | | | |
| | | 15:8 | UHEAD[15:8] | | | | | | | |
| | | 23:16 | UHEAD[23:16] | | | | | | | |
| | | 31:24 | UHEAD[31:24] | | | | | | | |
| 0x0370 | LCDC_HEOUADDR | 7:0 | UADDR[7:0] | | | | | | | |
| | | 15:8 | UADDR[15:8] | | | | | | | |
| | | 23:16 | UADDR[23:16] | | | | | | | |
| | | 31:24 | UADDR[31:24] | | | | | | | |
| 0x0374 | LCDC_HEOUCTRL | 7:0 | | | UDONEIEN | UADDIEN | UDSCRIEN | UDMAIEN | | UDFETCH |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0378 | LCDC_HEOUNEXT | 7:0 | UNEXT[7:0] | | | | | | | |
| | | 15:8 | UNEXT[15:8] | | | | | | | |
| | | 23:16 | UNEXT[23:16] | | | | | | | |
| | | 31:24 | UNEXT[31:24] | | | | | | | |
| 0x037C | LCDC_HEOVHEAD | 7:0 | VHEAD[7:0] | | | | | | | |
| | | 15:8 | VHEAD[15:8] | | | | | | | |
| | | 23:16 | VHEAD[23:16] | | | | | | | |
| | | 31:24 | VHEAD[31:24] | | | | | | | |
| 0x0380 | LCDC_HEOVADDR | 7:0 | VADDR[7:0] | | | | | | | |
| | | 15:8 | VADDR[15:8] | | | | | | | |
| | | 23:16 | VADDR[23:16] | | | | | | | |
| | | 31:24 | VADDR[31:24] | | | | | | | |
| 0x0384 | LCDC_HEOVCTRL | 7:0 | | | VDONEIEN | VADDIEN | VDSCRIEN | VDMAIEN | | VDFETCH |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0388 | LCDC_HEOVNEXT | 7:0 | VNEXT[7:0] | | | | | | | | |
| | | 15:8 | VNEXT[15:8] | | | | | | | | |
| | | 23:16 | VNEXT[23:16] | | | | | | | | |
| | | 31:24 | VNEXT[31:24] | | | | | | | | |
| 0x038C | LCDC_HEOCFG0 | 7:0 | BLENUV[1:0] | | BLEN[1:0] | | | | | | SIF |
| | | 15:8 | | | LOCKDIS | ROTDIS | | | | | DLBO |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x038C | LCDC_HEOCFG1 | 7:0 | RGBMODE[3:0] | | | | | | YUVEN | CLUTEN | |
| | | 15:8 | YUVMODE[3:0] | | | | | | CLUTMODE[1:0] | | |
| | | 23:16 | | | | DSCALEOPT | | | YUV422SWP | YUV422ROT | |
| | | 31:24 | | | | | | | | | |
| 0x0390 ... 0x0393 | Reserved | | | | | | | | | | |
| 0x0394 | LCDC_HEOCFG2 | 7:0 | XPOS[7:0] | | | | | | | | |
| | | 15:8 | | | | | XPOS[10:8] | | | | |
| | | 23:16 | YPOS[7:0] | | | | | | | | |
| | | 31:24 | | | | | YPOS[10:8] | | | | |
| 0x0398 | LCDC_HEOCFG3 | 7:0 | XSIZE[7:0] | | | | | | | | |
| | | 15:8 | | | | | XSIZE[10:8] | | | | |
| | | 23:16 | YSIZE[7:0] | | | | | | | | |
| | | 31:24 | | | | | YSIZE[10:8] | | | | |
| 0x039C | LCDC_HEOCFG4 | 7:0 | XMEMSIZE[7:0] | | | | | | | | |
| | | 15:8 | | | | | XMEMSIZE[10:8] | | | | |
| | | 23:16 | YMEMSIZE[7:0] | | | | | | | | |
| | | 31:24 | | | | | YMEMSIZE[10:8] | | | | |
| 0x03A0 | LCDC_HEOCFG5 | 7:0 | XSTRIDE[7:0] | | | | | | | | |
| | | 15:8 | XSTRIDE[15:8] | | | | | | | | |
| | | 23:16 | XSTRIDE[23:16] | | | | | | | | |
| | | 31:24 | XSTRIDE[31:24] | | | | | | | | |
| 0x03A4 | LCDC_HEOCFG6 | 7:0 | PSTRIDE[7:0] | | | | | | | | |
| | | 15:8 | PSTRIDE[15:8] | | | | | | | | |
| | | 23:16 | PSTRIDE[23:16] | | | | | | | | |
| | | 31:24 | PSTRIDE[31:24] | | | | | | | | |
| 0x03A8 | LCDC_HEOCFG7 | 7:0 | UVXSTRIDE[7:0] | | | | | | | | |
| | | 15:8 | UVXSTRIDE[15:8] | | | | | | | | |
| | | 23:16 | UVXSTRIDE[23:16] | | | | | | | | |
| | | 31:24 | UVXSTRIDE[31:24] | | | | | | | | |
| 0x03AC | LCDC_HEOCFG8 | 7:0 | UVPSTRIDE[7:0] | | | | | | | | |
| | | 15:8 | UVPSTRIDE[15:8] | | | | | | | | |
| | | 23:16 | UVPSTRIDE[23:16] | | | | | | | | |
| | | 31:24 | UVPSTRIDE[31:24] | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x03B0 | LCDC_HEOCFG9 | 7:0 | BDEF[7:0] | | | | | | | |
| | | 15:8 | GDEF[7:0] | | | | | | | |
| | | 23:16 | RDEF[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03B4 | LCDC_HEOCFG10 | 7:0 | BKEY[7:0] | | | | | | | |
| | | 15:8 | GKEY[7:0] | | | | | | | |
| | | 23:16 | RKEY[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03B8 | LCDC_HEOCFG11 | 7:0 | BMASK[7:0] | | | | | | | |
| | | 15:8 | GMASK[7:0] | | | | | | | |
| | | 23:16 | RMASK[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03BC | LCDC_HEOCFG12 | 7:0 | OVR | LAEN | GAEN | REVALPHA | ITER | ITER2BL | INV | CRKEY |
| | | 15:8 | | | | VIDPRI | | DSTKEY | REP | DMA |
| | | 23:16 | GA[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03C0 | LCDC_HEOCFG13 | 7:0 | XFACTOR[7:0] | | | | | | | |
| | | 15:8 | | | XFACTOR[13:8] | | | | | |
| | | 23:16 | YFACTOR[7:0] | | | | | | | |
| | | 31:24 | SCALEN | | YFACTOR[13:8] | | | | | |
| 0x03C4 | LCDC_HEOCFG14 | 7:0 | CSCRY[7:0] | | | | | | | |
| | | 15:8 | CSCRU[5:0] | | | | | | CSCRY[9:8] | |
| | | 23:16 | CSCRV[3:0] | | | | CSCRU[9:6] | | | |
| | | 31:24 | | CSCYOFF | CSCRV[9:4] | | | | | |
| 0x03C8 | LCDC_HEOCFG15 | 7:0 | CSCGY[7:0] | | | | | | | |
| | | 15:8 | CSCGU[5:0] | | | | | | CSCGY[9:8] | |
| | | 23:16 | CSCGV[3:0] | | | | CSCGU[9:6] | | | |
| | | 31:24 | | CSCUOFF | CSCGV[9:4] | | | | | |
| 0x03CC | LCDC_HEOCFG16 | 7:0 | CSCBY[7:0] | | | | | | | |
| | | 15:8 | CSCBU[5:0] | | | | | | CSCBY[9:8] | |
| | | 23:16 | CSCBV[3:0] | | | | CSCBU[9:6] | | | |
| | | 31:24 | | CSCVOFF | CSCBV[9:4] | | | | | |
| 0x03D0 | LCDC_HEOCFG17 | 7:0 | XPHI0COEFF0[7:0] | | | | | | | |
| | | 15:8 | XPHI0COEFF1[7:0] | | | | | | | |
| | | 23:16 | XPHI0COEFF2[7:0] | | | | | | | |
| | | 31:24 | XPHI0COEFF3[7:0] | | | | | | | |
| 0x03D4 | LCDC_HEOCFG18 | 7:0 | XPHI0COEFF4[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03D8 | LCDC_HEOCFG19 | 7:0 | XPHI1COEFF0[7:0] | | | | | | | |
| | | 15:8 | XPHI1COEFF1[7:0] | | | | | | | |
| | | 23:16 | XPHI1COEFF2[7:0] | | | | | | | |
| | | 31:24 | XPHI1COEFF3[7:0] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x03DC | LCDC_HEOCFG20 | 7:0 | XPHI1COEFF4[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03E0 | LCDC_HEOCFG21 | 7:0 | XPHI2COEFF0[7:0] | | | | | | | |
| | | 15:8 | XPHI2COEFF1[7:0] | | | | | | | |
| | | 23:16 | XPHI2COEFF2[7:0] | | | | | | | |
| | | 31:24 | XPHI2COEFF3[7:0] | | | | | | | |
| 0x03E4 | LCDC_HEOCFG22 | 7:0 | XPHI2COEFF4[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03E8 | LCDC_HEOCFG23 | 7:0 | XPHI3COEFF0[7:0] | | | | | | | |
| | | 15:8 | XPHI3COEFF1[7:0] | | | | | | | |
| | | 23:16 | XPHI3COEFF2[7:0] | | | | | | | |
| | | 31:24 | XPHI3COEFF3[7:0] | | | | | | | |
| 0x03EC | LCDC_HEOCFG24 | 7:0 | XPHI3COEFF4[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03F0 | LCDC_HEOCFG25 | 7:0 | XPHI4COEFF0[7:0] | | | | | | | |
| | | 15:8 | XPHI4COEFF1[7:0] | | | | | | | |
| | | 23:16 | XPHI4COEFF2[7:0] | | | | | | | |
| | | 31:24 | XPHI4COEFF3[7:0] | | | | | | | |
| 0x03F4 | LCDC_HEOCFG26 | 7:0 | XPHI4COEFF4[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03F8 | LCDC_HEOCFG27 | 7:0 | XPHI5COEFF0[7:0] | | | | | | | |
| | | 15:8 | XPHI5COEFF1[7:0] | | | | | | | |
| | | 23:16 | XPHI5COEFF2[7:0] | | | | | | | |
| | | 31:24 | XPHI5COEFF3[7:0] | | | | | | | |
| 0x03FC | LCDC_HEOCFG28 | 7:0 | XPHI5COEFF4[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0400 | LCDC_HEOCFG29 | 7:0 | XPHI6COEFF0[7:0] | | | | | | | |
| | | 15:8 | XPHI6COEFF1[7:0] | | | | | | | |
| | | 23:16 | XPHI6COEFF2[7:0] | | | | | | | |
| | | 31:24 | XPHI6COEFF3[7:0] | | | | | | | |
| 0x0404 | LCDC_HEOCFG30 | 7:0 | XPHI6COEFF4[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0408 | LCDC_HEOCFG31 | 7:0 | XPHI7COEFF0[7:0] | | | | | | | |
| | | 15:8 | XPHI7COEFF1[7:0] | | | | | | | |
| | | 23:16 | XPHI7COEFF2[7:0] | | | | | | | |
| | | 31:24 | XPHI7COEFF3[7:0] | | | | | | | |
| 0x040C | LCDC_HEOCFG32 | 7:0 | XPHI7COEFF4[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0410 | LCDC_HEOCFG33 | 7:0 | YPHI0COEFF0[7:0] | | | | | | | |
| | | 15:8 | YPHI0COEFF1[7:0] | | | | | | | |
| | | 23:16 | YPHI0COEFF2[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0414 | LCDC_HEOCFG34 | 7:0 | YPHI1COEFF0[7:0] | | | | | | | |
| | | 15:8 | YPHI1COEFF1[7:0] | | | | | | | |
| | | 23:16 | YPHI1COEFF2[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0418 | LCDC_HEOCFG35 | 7:0 | YPHI2COEFF0[7:0] | | | | | | | |
| | | 15:8 | YPHI2COEFF1[7:0] | | | | | | | |
| | | 23:16 | YPHI2COEFF2[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x041C | LCDC_HEOCFG36 | 7:0 | YPHI3COEFF0[7:0] | | | | | | | |
| | | 15:8 | YPHI3COEFF1[7:0] | | | | | | | |
| | | 23:16 | YPHI3COEFF2[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0420 | LCDC_HEOCFG37 | 7:0 | YPHI4COEFF0[7:0] | | | | | | | |
| | | 15:8 | YPHI4COEFF1[7:0] | | | | | | | |
| | | 23:16 | YPHI4COEFF2[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0424 | LCDC_HEOCFG38 | 7:0 | YPHI5COEFF0[7:0] | | | | | | | |
| | | 15:8 | YPHI5COEFF1[7:0] | | | | | | | |
| | | 23:16 | YPHI5COEFF2[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0428 | LCDC_HEOCFG39 | 7:0 | YPHI6COEFF0[7:0] | | | | | | | |
| | | 15:8 | YPHI6COEFF1[7:0] | | | | | | | |
| | | 23:16 | YPHI6COEFF2[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x042C | LCDC_HEOCFG40 | 7:0 | YPHI7COEFF0[7:0] | | | | | | | |
| | | 15:8 | YPHI7COEFF1[7:0] | | | | | | | |
| | | 23:16 | YPHI7COEFF2[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0430 | LCDC_HEOCFG41 | 7:0 | | | | | | XPHIDEF[2:0] | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | YPHIDEF[2:0] | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0434 ... 0x053F | Reserved | | | | | | | | | |
| 0x0540 | LCDC_PPCHER | 7:0 | | | | | | A2QEN | UPDATEEN | CHEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0544 | LCDC_PPCHDR | 7:0 | | | | | | | | CHDIS |
| | | 15:8 | | | | | | | | CHRST |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0548 | LCDC_PPCHSR | 7:0 | | | | | | A2QSR | UPDATESR | CHSR |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x054C | LCDC_PPIER | 7:0 | | | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0550 | LCDC_PPIDR | 7:0 | | | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0554 | LCDC_PPIMR | 7:0 | | | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0558 | LCDC_PPISR | 7:0 | | | DONE | ADD | DSCR | DMA | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x055C | LCDC_PPHEAD | 7:0 | HEAD[5:0] | | | | | | | |
| | | 15:8 | HEAD[13:6] | | | | | | | |
| | | 23:16 | HEAD[21:14] | | | | | | | |
| | | 31:24 | HEAD[29:22] | | | | | | | |
| 0x0560 | LCDC_PPADDR | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0x0564 | LCDC_PPCTRL | 7:0 | | | DONEIEN | ADDIEN | DSCRIEN | DMAIEN | | DFETCH |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0568 | LCDC_PPNEXT | 7:0 | NEXT[7:0] | | | | | | | |
| | | 15:8 | NEXT[15:8] | | | | | | | |
| | | 23:16 | NEXT[23:16] | | | | | | | |
| | | 31:24 | NEXT[31:24] | | | | | | | |
| 0x056C | LCDC_PPCFG0 | 7:0 | | | BLEN[1:0] | | | | | SIF |
| | | 15:8 | | | | | | | | DLBO |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0570 | LCDC_PPCFG1 | 7:0 | | | | ITUBT601 | | PPMODE[2:0] | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0574 | LCDC_PPCFG2 | 7:0 | XSTRIDE[7:0] | | | | | | | |
| | | 15:8 | XSTRIDE[15:8] | | | | | | | |
| | | 23:16 | XSTRIDE[23:16] | | | | | | | |
| | | 31:24 | XSTRIDE[31:24] | | | | | | | |
| 0x0578 | LCDC_PPCFG3 | 7:0 | CSCYR[7:0] | | | | | | | |
| | | 15:8 | CSCYG[5:0] | | | | | | CSCYR[9:8] | |
| | | 23:16 | CSCYB[3:0] | | | | CSCYG[9:6] | | | |
| | | 31:24 | | CSCYOFF | CSCYB[9:4] | | | | | |
| 0x057C | LCDC_PPCFG4 | 7:0 | CSCUR[7:0] | | | | | | | |
| | | 15:8 | CSCUG[5:0] | | | | | | CSCUR[9:8] | |
| | | 23:16 | CSCUB[3:0] | | | | CSCUG[9:6] | | | |
| | | 31:24 | | CSCUOFF | CSCUB[9:4] | | | | | |
| 0x0580 | LCDC_PPCFG5 | 7:0 | CSCVR[7:0] | | | | | | | |
| | | 15:8 | CSCVG[5:0] | | | | | | CSCVR[9:8] | |
| | | 23:16 | CSCVB[3:0] | | | | CSCVG[9:6] | | | |
| | | 31:24 | | CSCVOFF | CSCVB[9:4] | | | | | |
| 0x0584 ... 0x05FF | Reserved | | | | | | | | | |
| 0x0600 | LCDC_BASECLUT0 | 7:0 | BCLUT[7:0] | | | | | | | |
| | | 15:8 | GCLUT[7:0] | | | | | | | |
| | | 23:16 | RCLUT[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| ... | | | | | | | | | | |
| 0x09FC | LCDC_BASECLUT255 | 7:0 | BCLUT[7:0] | | | | | | | |
| | | 15:8 | GCLUT[7:0] | | | | | | | |
| | | 23:16 | RCLUT[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0A00 | LCDC_OVR1CLUT0 | 7:0 | BCLUT[7:0] | | | | | | | |
| | | 15:8 | GCLUT[7:0] | | | | | | | |
| | | 23:16 | RCLUT[7:0] | | | | | | | |
| | | 31:24 | ACLUT[7:0] | | | | | | | |
| ... | | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0DFC | LCDC_OVR1CLUT255 | 7:0 | BCLUT[7:0] | | | | | | | |
| | | 15:8 | GCLUT[7:0] | | | | | | | |
| | | 23:16 | RCLUT[7:0] | | | | | | | |
| | | 31:24 | ACLUT[7:0] | | | | | | | |
| 0x0E00 | LCDC_OVR2CLUT0 | 7:0 | BCLUT[7:0] | | | | | | | |
| | | 15:8 | GCLUT[7:0] | | | | | | | |
| | | 23:16 | RCLUT[7:0] | | | | | | | |
| | | 31:24 | ACLUT[7:0] | | | | | | | |
| ... | | | | | | | | | | |
| 0x11FC | LCDC_OVR2CLUT255 | 7:0 | BCLUT[7:0] | | | | | | | |
| | | 15:8 | GCLUT[7:0] | | | | | | | |
| | | 23:16 | RCLUT[7:0] | | | | | | | |
| | | 31:24 | ACLUT[7:0] | | | | | | | |
| 0x1200 | LCDC_HEOCLUT0 | 7:0 | BCLUT[7:0] | | | | | | | |
| | | 15:8 | GCLUT[7:0] | | | | | | | |
| | | 23:16 | RCLUT[7:0] | | | | | | | |
| | | 31:24 | ACLUT[7:0] | | | | | | | |
| ... | | | | | | | | | | |
| 0x15FC | LCDC_HEOCLUT255 | 7:0 | BCLUT[7:0] | | | | | | | |
| | | 15:8 | GCLUT[7:0] | | | | | | | |
| | | 23:16 | RCLUT[7:0] | | | | | | | |
| | | 31:24 | ACLUT[7:0] | | | | | | | |

### 38.7.1 LCD Controller Configuration Register 0

**Name:** LCDC_LCDCFG0
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CLKDIV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | CGDISPP | | CGDISHEO | CGDISOVR2 | CGDISOVR1 | CGDISBASE |
| Access | | | R/W | | R/W | R/W | R/W | R/W |
| Reset | | | 0 | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CLKPWMSEL | CLKSEL | | CLKPOL |
| Access | | | | | R/W | R/W | | R/W |
| Reset | | | | | 0 | 0 | | 0 |

**Bits 23:16 – CLKDIV[7:0]** LCD Controller Clock Divider
8-bit width clock divider for pixel clock (LCDPCLK). The pixel clock period formula is:

LCDPCLK = source clock / (CLKDIV+2)

where source clock is the system clock when CLKSEL is written to ‘0’, and 2x system_clock when CLKSEL is written to ‘1’.

**Bit 13 – CGDISPP** Clock Gating Disable Control for the Post Processing Layer

| Value | Description |
|---|---|
| 0 | Automatic Clock Gating is enabled for the Post Processing Layer. |
| 1 | Clock is running continuously. |

**Bit 11 – CGDISHEO** Clock Gating Disable Control for the High-End Overlay

| Value | Description |
|---|---|
| 0 | Automatic Clock Gating is enabled for the High-End Overlay Layer. |
| 1 | Clock is running continuously. |

**Bit 10 – CGDISOVR2** Clock Gating Disable Control for the Overlay 2 Layer

| Value | Description |
|---|---|
| 0 | Automatic Clock Gating is enabled for the Overlay 2 Layer. |
| 1 | Clock is running continuously. |

**Bit 9 – CGDISOVR1** Clock Gating Disable Control for the Overlay 1 Layer

| Value | Description |
|---|---|
| 0 | Automatic Clock Gating is enabled for the Overlay 1 Layer. |
| 1 | Clock is running continuously. |

**Bit 8 – CGDISBASE** Clock Gating Disable Control for the Base Layer

| Value | Description |
|---|---|
| 0 | Automatic Clock Gating is enabled for the Base Layer. |
| 1 | Clock is running continuously. |

**Bit 3 – CLKPWMSEL** LCD Controller PWM Clock Source Selection

| Value | Description |
|---|---|
| 0 | The slow clock is selected and feeds the PWM module. |
| 1 | The system clock is selected and feeds the PWM module. |

**Bit 2 – CLKSEL** LCD Controller Clock Source Selection

| Value | Description |
|---|---|
| 0 | The asynchronous output stage of the LCD controller is fed by the System Clock. |
| 1 | The asynchronous output state of the LCD controller is fed by the 2x System Clock. |

**Bit 0 – CLKPOL** LCD Controller Clock Polarity

| Value | Description |
|---|---|
| 0 | Data/Control signals are launched on the rising edge of the pixel clock. |
| 1 | Data/Control signals are launched on the falling edge of the pixel clock. |

### 38.7.2 LCD Controller Configuration Register 1

**Name:** LCDC_LCDCFG1
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | VSPW[9:8] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | VSPW[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | HSPW[9:8] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | HSPW[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 25:16 – VSPW[9:0]** Vertical Synchronization Pulse Width
Width of the LCDVSYNC pulse, given in number of lines. Width is (VSPW+1) lines.

**Bits 9:0 – HSPW[9:0]** Horizontal Synchronization Pulse Width
Width of the LCDHSYNC pulse, given in pixel clock cycles. Width is (HSPW+1) LCDPCLK cycles.

### 38.7.3 LCD Controller Configuration Register 2

**Name:** LCDC_LCDCFG2
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | VBPW[9:8] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | VBPW[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | VFPW[9:8] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | VFPW[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 25:16 – VBPW[9:0]** Vertical Back Porch Width
This field indicates the number of lines at the beginning of the Frame. The blanking interval is equal to VBPW lines.

**Bits 9:0 – VFPW[9:0]** Vertical Front Porch Width
This field indicates the number of lines at the end of the Frame. The blanking interval is equal to (VFPW +1) lines.

### 38.7.4 LCD Controller Configuration Register 3

**Name:** LCDC_LCDCFG3
**Offset:** 0x0C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | HBPW[9:8] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | HBPW[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | HFPW[9:8] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | HFPW[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 25:16 – HBPW[9:0]** Horizontal Back Porch Width
Number of pixel clock cycles inserted at the beginning of the line. The interval is equal to (HBPW+1) LCDPCLK cycles.

**Bits 9:0 – HFPW[9:0]** Horizontal Front Porch Width
Number of pixel clock cycles inserted at the end of the active line. The interval is equal to (HFPW+1) LCDPCLK cycles.

### 38.7.5 LCD Controller Configuration Register 4

**Name:** LCDC_LCDCFG4
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | RPF[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RPF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PPL[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PPL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 26:16 – RPF[10:0]** Number of Active Row Per Frame
Number of active lines in the frame. The frame height is equal to (RPF+1) lines.

**Bits 10:0 – PPL[10:0]** Number of Pixels Per Line
Number of pixels in the frame. The number of active pixels in the frame is equal to (PPL+1) pixels.

### 38.7.6 LCD Controller Configuration Register 5

**Name:** LCDC_LCDCFG5
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GUARDTIME[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | VSPHO | VSPSU | | PP | MODE[1:0] | |
| Access | | | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | 0 | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DISPDLY | DITHER | | DISPPOL | VSPDLYE | VSPDLYS | VSPOL | HSPOL |
| Access | R/W | R/W | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – GUARDTIME[7:0]** LCD DISPLAY Guard Time
Number of frames inserted during startup before LCDDISP assertion.

Number of frames inserted after LCDDISP reset.

**Bit 13 – VSPHO** LCD Controller Vertical synchronization Pulse Hold Configuration

| Value | Description |
|---|---|
| 0 | The vertical synchronization pulse is asserted synchronously with horizontal pulse edge. |
| 1 | The vertical synchronization pulse is held active one pixel clock cycle after the horizontal pulse. |

**Bit 12 – VSPSU** LCD Controller Vertical synchronization Pulse Setup Configuration

| Value | Description |
|---|---|
| 0 | The vertical synchronization pulse is asserted synchronously with horizontal pulse edge. |
| 1 | The vertical synchronization pulse is asserted one pixel clock cycle before the horizontal pulse. |

**Bit 10 – PP** Post Processing Enable

| Value | Description |
|---|---|
| 0 | The blended pixel is pushed into the output FIFO. |

| Value | Description |
|---|---|
| 1 | The blended pixel is written back to memory, the post-processing stage is enabled. |

**Bits 9:8 – MODE[1:0]** LCD Controller Output Mode

| Value | Name | Description |
|---|---|---|
| 0 | OUTPUT_12BPP | LCD Output mode is set to 12 bits per pixel |
| 1 | OUTPUT_16BPP | LCD Output mode is set to 16 bits per pixel |
| 2 | OUTPUT_18BPP | LCD Output mode is set to 18 bits per pixel |
| 3 | OUTPUT_24BPP | LCD Output mode is set to 24 bits per pixel |

**Bit 7 – DISPDLY** LCD Controller Display Power Signal Synchronization

| Value | Description |
|---|---|
| 0 | The LCDDISP signal is asserted synchronously with the second active edge of the horizontal pulse. |
| 1 | The LCDDISP signal is asserted asynchronously with both edges of the horizontal pulse. |

**Bit 6 – DITHER** LCD Controller Dithering

| Value | Description |
|---|---|
| 0 | Dithering logical unit is disabled |
| 1 | Dithering logical unit is activated |

**Bit 4 – DISPPOL** Display Signal Polarity

| Value | Description |
|---|---|
| 0 | Active High |
| 1 | Active Low |

**Bit 3 – VSPDLYE** Vertical Synchronization Pulse End

| Value | Description |
|---|---|
| 0 | The second active edge of the Vertical synchronization pulse is synchronous with the second edge of the horizontal pulse. |
| 1 | The second active edge of the Vertical synchronization pulse is synchronous with the first edge of the horizontal pulse. |

**Bit 2 – VSPDLYS** Vertical Synchronization Pulse Start

| Value | Description |
|---|---|
| 0 | The first active edge of the Vertical synchronization pulse is synchronous with the second edge of the horizontal pulse. |
| 1 | The first active edge of the Vertical synchronization pulse is synchronous with the first edge of the horizontal pulse. |

**Bit 1 – VSPOL** Vertical Synchronization Pulse Polarity

| Value | Description |
|---|---|
| 0 | Active High |
| 1 | Active Low |

**Bit 0 – HSPOL** Horizontal Synchronization Pulse Polarity

| Value | Description |
|---|---|
| 0 | Active High |
| 1 | Active Low |

### 38.7.7 LCD Controller Configuration Register 6

**Name:** LCDC_LCDCFG6
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PWMCVAL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | PWMPOL | | PWMPS[2:0] | | |
| Access | | | | R/W | | R/W | R/W | R/W |
| Reset | | | | 0 | | 0 | 0 | 0 |

**Bits 15:8 – PWMCVAL[7:0]** LCD Controller PWM Compare Value
PWM compare value. Used to adjust the analog value obtained after an external filter to control the contrast of the display.

**Bit 4 – PWMPOL** LCD Controller PWM Signal Polarity
This bit defines the polarity of the PWM output signal.

| Value | Description |
|---|---|
| 0 | The output pulses are low level. |
| 1 | The output pulses are high level (the output is high whenever the value in the counter is less than value CVAL). |

**Bits 2:0 – PWMPS[2:0]** PWM Clock Prescaler
Selects the configuration of the counter prescaler module.

| Value | Name | Description |
|---|---|---|
| 000 | DIV_1 | The counter advances at a rate of $f_{COUNTER} = f_{PWM_SELECTED_CLOCK}$ |
| 001 | DIV_2 | The counter advances at a rate of $f_{COUNTER} = f_{PWM_SELECTED_CLOCK/2}$ |
| 010 | DIV_4 | The counter advances at a rate of $f_{COUNTER} = f_{PWM_SELECTED_CLOCK/4}$ |
| 011 | DIV_8 | The counter advances at a rate of $f_{COUNTER} = f_{PWM_SELECTED_CLOCK/8}$ |
| 100 | DIV_16 | The counter advances at a rate of $f_{COUNTER} = f_{PWM_SELECTED_CLOCK/16}$ |
| 101 | DIV_32 | The counter advances at a of rate $f_{COUNTER} = f_{PWM_SELECTED_CLOCK/32}$ |

| Value | Name | Description |
|---|---|---|
| 110 | DIV_64 | The counter advances at a of rate $f_{COUNTER} = f_{PWM_SELECTED_CLOCK/64}$ |

### 38.7.8 LCD Controller Enable Register

**Name:** LCDC_LCDEN
**Offset:** 0x20
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PWMEN | DISPEN | SYNCEN | CLKEN |
| Access | | | | | W | W | | W |
| Reset | | | | | – | – | – | – |

**Bit 3 – PWMEN** LCD Controller Pulse Width Modulation Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | PWM is enabled. |

**Bit 2 – DISPEN** LCD Controller DISP Signal Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | LCDDISP signal is generated. |

**Bit 1 – SYNCEN** LCD Controller Horizontal and Vertical Synchronization Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Both horizontal and vertical synchronization (LCDVSYNC and LCDHSYNC) signals are generated. |

**Bit 0 – CLKEN** LCD Controller Pixel Clock Enable

| Value | Description |
|---|---|
| 0 | No effect |

| Value | Description |
|---|---|
| 1 | Pixel clock logical unit is activated. |

### 38.7.9 LCD Controller Disable Register

**Name:** LCDC_LCDDIS
**Offset:** 0x24
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PWMRST | DISPRST | SYNCRST | CLKRST |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PWMDIS | DISPDIS | SYNCDIS | CLKDIS |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | – | – | – | – |

**Bit 11 – PWMRST** LCD Controller PWM Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the PWM module. The duty cycle may be violated. |

**Bit 10 – DISPRST** LCD Controller DISP Signal Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the DISP signal. |

**Bit 9 – SYNCRST** LCD Controller Horizontal and Vertical Synchronization Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the timing engine. The horizontal and vertical pulse widths are both violated. |

**Bit 8 – CLKRST** LCD Controller Clock Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the pixel clock generator module. The pixel clock duty cycle may be violated. |

**Bit 3 – PWMDIS** LCD Controller Pulse Width Modulation Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the pulse width modulation signal. |

**Bit 2 – DISPDIS** LCD Controller DISP Signal Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the DISP signal. |

**Bit 1 – SYNCDIS** LCD Controller Horizontal and Vertical Synchronization Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the synchronization signals after the end of the frame. |

**Bit 0 – CLKDIS** LCD Controller Pixel Clock Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the pixel clock. |

### 38.7.10 LCD Controller Status Register

**Name:** LCDC_LCDSR
**Offset:** 0x28
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | SIPSTS | PWMSTS | DISPSTS | LCDSTS | CLKSTS |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bit 4 – SIPSTS** Synchronization In Progress

| Value | Description |
|---|---|
| 0 | Clock domain synchronization is terminated. |
| 1 | Synchronization is in progress. Access to the registers LCDC_LCDCCFG[0..6], LCDC_LCDEN and LCDC_LCDDIS has no effect. |

**Bit 3 – PWMSTS** LCD Controller PWM Signal Status

| Value | Description |
|---|---|
| 0 | PWM is disabled. |
| 1 | PWM signal is activated. |

**Bit 2 – DISPSTS** LCD Controller DISP Signal Status

| Value | Description |
|---|---|
| 0 | DISP is disabled. |
| 1 | DISP signal is activated. |

**Bit 1 – LCDSTS** LCD Controller Synchronization status

| Value | Description |
|---|---|
| 0 | Timing engine is disabled. |

| Value | Description |
|---|---|
| 1 | Timing engine is running. |

**Bit 0 – CLKSTS** Clock Status

| Value | Description |
|---|---|
| 0 | Pixel clock is disabled. |
| 1 | Pixel clock is running. |

### 38.7.11 LCD Controller Interrupt Enable Register

**Name:** LCDC_LCDIER
**Offset:** 0x2C
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | PPIE | | HEOIE | OVR2IE | OVR1IE | BASEIE |
| Access | | | W | | W | W | W | W |
| Reset | | | – | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | FIFOERRIE | | DISPIE | DISIE | SOFIE |
| Access | | | | W | | W | W | W |
| Reset | | | | – | | – | – | – |

**Bit 13 – PPIE** Post Processing Interrupt Enable

**Bit 11 – HEOIE** High-End Overlay Interrupt Enable

**Bit 10 – OVR2IE** Overlay 2 Interrupt Enable

**Bit 9 – OVR1IE** Overlay 1 Interrupt Enable

**Bit 8 – BASEIE** Base Layer Interrupt Enable

**Bit 4 – FIFOERRIE** Output FIFO Error Interrupt Enable

**Bit 2 – DISPIE** Powerup/Powerdown Sequence Terminated Interrupt Enable

**Bit 1 – DISIE** LCD Disable Interrupt Enable

**Bit 0 – SOFIE** Start of Frame Interrupt Enable

© 2018 Microchip Technology Inc.

### 38.7.12 LCD Controller Interrupt Disable Register

**Name:** LCDC_LCDIDR
**Offset:** 0x30
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | PPID | | HEOID | OVR2ID | OVR1ID | BASEID |
| Access | | | W | | W | W | W | W |
| Reset | | | – | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | FIFOERRID | | DISPID | DISID | SOFID |
| Access | | | | W | | W | W | W |
| Reset | | | | – | | – | – | – |

**Bit 13 – PPID** Post Processing Interrupt Disable

**Bit 11 – HEOID** High-End Overlay Interrupt Disable

**Bit 10 – OVR2ID** Overlay 2 Interrupt Disable

**Bit 9 – OVR1ID** Overlay 1 Interrupt Disable

**Bit 8 – BASEID** Base Layer Interrupt Disable

**Bit 4 – FIFOERRID** Output FIFO Error Interrupt Disable

**Bit 2 – DISPID** Powerup/Powerdown Sequence Terminated Interrupt Disable

**Bit 1 – DISID** LCD Disable Interrupt Disable

**Bit 0 – SOFID** Start of Frame Interrupt Disable

### 38.7.13 LCD Controller Interrupt Mask Register

**Name:** LCDC_LCDIMR
**Offset:** 0x34
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:
0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | PPIM | | HEOIM | OVR2IM | OVR1IM | BASEIM |
| Access | | | R | | R | R | R | R |
| Reset | | | 0 | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | FIFOERRIM | | DISPIM | DISIM | SOFIM |
| Access | | | | R | | R | R | R |
| Reset | | | | 0 | | 0 | 0 | 0 |

**Bit 13 – PPIM** Post Processing Interrupt Mask

**Bit 11 – HEOIM** High-End Overlay Interrupt Mask

**Bit 10 – OVR2IM** Overlay 2 Interrupt Mask

**Bit 9 – OVR1IM** Overlay 1 Interrupt Mask

**Bit 8 – BASEIM** Base Layer Interrupt Mask

**Bit 4 – FIFOERRIM** Output FIFO Error Interrupt Mask

**Bit 2 – DISPIM** Powerup/Powerdown Sequence Terminated Interrupt Mask

**Bit 1 – DISIM** LCD Disable Interrupt Mask

**Bit 0 – SOFIM** Start of Frame Interrupt Mask

### 38.7.14 LCD Controller Interrupt Status Register

**Name:** LCDC_LCDISR
**Offset:** 0x38
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | PP | | HEO | OVR2 | OVR1 | BASE |
| Access | | | R | | R | R | R | R |
| Reset | | | 0 | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | FIFOERR | | DISP | DIS | SOF |
| Access | | | | R | | R | R | R |
| Reset | | | | 0 | | 0 | 0 | 0 |

**Bit 13 – PP** Post Processing Raw Interrupt Status

| Value | Description |
|---|---|
| 0 | No Post Processing interrupt detected since last read of LCDC_PPISR |
| 1 | Indicates that Post Processing interrupt is pending. This flag is reset as soon as the LCDC_PPISR is read. |

**Bit 11 – HEO** High-End Overlay Raw Interrupt Status

| Value | Description |
|---|---|
| 0 | No High-End layer interrupt detected since last read of LCDC_HEOISR. |
| 1 | Indicates that a High-End layer interrupt is pending. This flag is reset as soon as the LCDC_HEOISR is read. |

**Bit 10 – OVR2** Overlay 2 Raw Interrupt Status

| Value | Description |
|---|---|
| 0 | No Overlay 2 layer interrupt detected since last read of LCDC_OVR2ISR. |
| 1 | Indicates that an Overlay 2 layer interrupt is pending. This flag is reset as soon as the LCDC_OVR2ISR is read. |

**Bit 9 – OVR1** Overlay 1 Raw Interrupt Status

| Value | Description |
|---|---|
| 0 | No Overlay 1 layer interrupt detected since last read of LCDC_OVR1ISR. |
| 1 | Indicates that an Overlay 1 layer interrupt is pending. This flag is reset as soon as the LCDC_OVR1ISR is read. |

**Bit 8 – BASE** Base Layer Raw Interrupt Status

| Value | Description |
|---|---|
| 0 | No base layer interrupt detected since last read of LCDC_BASEISR. |
| 1 | Indicates that a base layer interrupt is pending. This flag is reset as soon as the LCDC_BASEISR is read. |

**Bit 4 – FIFOERR** Output FIFO Error

| Value | Description |
|---|---|
| 0 | No underflow has occurred in the output FIFO since last read of LCDC_LCDISR. |
| 1 | Indicates that an underflow has occurred in the output FIFO. This flag is reset after a read operation. |

**Bit 2 – DISP** Powerup/Powerdown Sequence Terminated Interrupt Status

| Value | Description |
|---|---|
| 0 | Powerup sequence or powerdown sequence has not yet terminated. |
| 1 | Indicates the powerup sequence or powerdown sequence has terminated. This flag is reset after a read operation. |

**Bit 1 – DIS** LCD Disable Interrupt Status

| Value | Description |
|---|---|
| 0 | Horizontal and vertical timing generator has not yet been disabled. |
| 1 | Indicates that the horizontal and vertical timing generator has been disabled. This flag is reset after a read operation. |

**Bit 0 – SOF** Start of Frame Interrupt Status

| Value | Description |
|---|---|
| 0 | No detection since last read of LCDC_LCDISR. |
| 1 | Indicates that a start of frame event has been detected. This flag is reset after a read operation. |

### 38.7.15 LCD Controller Attribute Register

**Name:** LCDC_ATTR
**Offset:** 0x3C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | PPA2Q | | HEOA2Q | OVR2A2Q | OVR1A2Q | BASEA2Q |
| Access | | | W | | W | W | W | W |
| Reset | | | – | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | PP | | HEO | OVR2 | OVR1 | BASE |
| Access | | | W | | W | W | W | W |
| Reset | | | – | | – | – | – | – |

**Bit 13 – PPA2Q** Post-Processing Update Add To Queue

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Add the descriptor pointed to by the LCDC_PPHEAD register to the descriptor list. |

**Bit 11 – HEOA2Q** High-End Overlay Update Add To Queue

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Add the descriptor pointed to by the LCDC_HEOHEAD register to the descriptor list. |

**Bit 10 – OVR2A2Q** Overlay 2 Update Add to Queue

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Add the descriptor pointed to by the LCDC_OVR2HEAD register to the descriptor list. |

**Bit 9 – OVR1A2Q** Overlay 1 Update Add To Queue

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Add the descriptor pointed to by the LCDC_OVR1HEAD register to the descriptor list. |

**Bit 8 – BASEA2Q** Base Layer Update Add To Queue

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Add the descriptor pointed to by the LCDC_BASEHEAD register to the descriptor list. |

**Bit 5 – PP** Post-Processing Update Attribute

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Update the PP window attribute. |

**Bit 3 – HEO** High-End Overlay Update Attribute

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Update the HEO window attribute. |

**Bit 2 – OVR2** Overlay 2 Update Attribute

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Update the OVR2 window attribute. |

**Bit 1 – OVR1** Overlay 1 Update Attribute

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Update the OVR1 window attribute. |

**Bit 0 – BASE** Base Layer Update Attribute

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Update the BASE window attributes. |

### 38.7.16 Base Layer Channel Enable Register

**Name:** LCDC_BASECHER
**Offset:** 0x00000040
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | A2QEN | UPDATEEN | CHEN |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

**Bit 2 – A2QEN** Add To Queue Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Indicates that a valid descriptor has been written to memory, its memory location should be written to the DMA head pointer. The A2QSR status bit is set to one, and it is reset by hardware as soon as the descriptor pointed to by the DMA head pointer is added to the list. |

**Bit 1 – UPDATEEN** Update Overlay Attributes Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Updates windows attributes on the next start of frame. |

**Bit 0 – CHEN** Channel Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Enables the DMA channel |

### 38.7.17 Base Layer Channel Disable Register

**Name:** LCDC_BASECHDR
**Offset:** 0x00000040
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 8 – CHRST** Channel Reset

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Resets the layer immediately. The frame is aborted. |

**Bit 0 – CHDIS** Channel Disable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Disables the layer at the end of the current frame. The frame is completed. |

### 38.7.18 Base Layer Channel Status Register

**Name:** LCDC_BASECHSR
**Offset:** 0x00000048
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | A2QSR | UPDATESR | CHSR |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – A2QSR** Add To Queue Status

| Value | Description |
|---|---|
| 0 | Add to queue not pending |
| 1 | Add to queue pending |

**Bit 1 – UPDATESR** Update Overlay Attributes In Progress Status

| Value | Description |
|---|---|
| 0 | No update pending |
| 1 | Overlay attributes will be updated on the next frame |

**Bit 0 – CHSR** Channel Status

| Value | Description |
|---|---|
| 0 | Layer disabled |
| 1 | Layer enabled |

### 38.7.19 Base Layer Interrupt Enable Register

**Name:** LCDC_BASEIER
**Offset:** 0x0000004C
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | OVR | DONE | ADD | DSCR | DMA | | |
| Access | | W | W | W | W | W | | |
| Reset | | – | – | – | – | – | | |

**Bit 6 – OVR** Overflow Interrupt Enable

**Bit 5 – DONE** End of List Interrupt Enable

**Bit 4 – ADD** Head Descriptor Loaded Interrupt Enable

**Bit 3 – DSCR** Descriptor Loaded Interrupt Enable

**Bit 2 – DMA** End of DMA Transfer Interrupt Enable

### 38.7.20 Base Layer Interrupt Disable Register

**Name:** LCDC_BASEIDR
**Offset:** 0x00000050
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | OVR | DONE | ADD | DSCR | DMA | | |
| Access | | W | W | W | W | W | | |
| Reset | | – | – | – | – | – | | |

**Bit 6 – OVR** Overflow Interrupt Disable

**Bit 5 – DONE** End of List Interrupt Disable

**Bit 4 – ADD** Head Descriptor Loaded Interrupt Disable

**Bit 3 – DSCR** Descriptor Loaded Interrupt Disable

**Bit 2 – DMA** End of DMA Transfer Interrupt Disable

### 38.7.21 Base Layer Interrupt Mask Register

**Name:** LCDC_BASEIMR
**Offset:** 0x00000054
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:
0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | OVR | DONE | ADD | DSCR | DMA | | |
| Access | | R | R | R | R | R | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | | |

**Bit 6 – OVR** Overflow Interrupt Mask

**Bit 5 – DONE** End of List Interrupt Mask

**Bit 4 – ADD** Head Descriptor Loaded Interrupt Mask

**Bit 3 – DSCR** Descriptor Loaded Interrupt Mask

**Bit 2 – DMA** End of DMA Transfer Interrupt Mask

### 38.7.22 Base Layer Interrupt Status Register

**Name:** LCDC_BASEISR
**Offset:** 0x00000058
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | OVR | DONE | ADD | DSCR | DMA | | |
| Access | | R | R | R | R | R | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | | |

**Bit 6 – OVR** Overflow Detected

| Value | Description |
|---|---|
| 0 | No overflow occurred since last read of LCDC_BASEISR |
| 1 | An overflow occurred. This flag is reset after a read operation. |

**Bit 5 – DONE** End of List Detected

| Value | Description |
|---|---|
| 0 | No End of List condition occurred since last read of LCDC_BASEISR |
| 1 | End of List condition has occurred. This flag is reset after a read operation. |

**Bit 4 – ADD** Head Descriptor Loaded

| Value | Description |
|---|---|
| 0 | No descriptor has been loaded since last read of LCDC_BASEISR |
| 1 | The descriptor pointed to by the LCDC_BASEHEAD register has been loaded successfully. This flag is reset after a read operation. |

**Bit 3 – DSCR** DMA Descriptor Loaded

| Value | Description |
|---|---|
| 0 | No descriptor has been loaded since last read of LCDC_BASEISR |

| Value | Description |
|---|---|
| 1 | A descriptor has been loaded successfully. This flag is reset after a read operation. |

**Bit 2 – DMA** End of DMA Transfer

| Value | Description |
|---|---|
| 0 | No end of DMA transfer has been detected since last read of LCDC_BASEISR |
| 1 | End of Transfer has been detected. This flag is reset after a read operation. |

### 38.7.23 Base DMA Head Register

**Name:** LCDC_BASEHEAD
**Offset:** 0x0000005C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – HEAD[29:0]** DMA Head Pointer
The Head Pointer points to a new descriptor.

### 38.7.24 Base DMA Address Register

**Name:** LCDC_BASEADDR
**Offset:** 0x00000060
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** DMA Transfer Start Address
Frame buffer base address

### 38.7.25 Base DMA Control Register

**Name:** LCDC_BASECTRL
**Offset:** 0x00000064
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DONEIEN | ADDIEN | DSCRIEN | DMAIEN | LFETCH | DFETCH |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 5 – DONEIEN** End of List Interrupt Enable

| Value | Description |
|---|---|
| 0 | End of list interrupt is disabled |
| 1 | End of list interrupt is enabled |

**Bit 4 – ADDIEN** Add Head Descriptor to Queue Interrupt Enable

| Value | Description |
|---|---|
| 0 | Transfer descriptor added to queue interrupt is enabled |
| 1 | Transfer descriptor added to queue interrupt is enabled |

**Bit 3 – DSCRIEN** Descriptor Loaded Interrupt Enable

| Value | Description |
|---|---|
| 0 | Transfer descriptor loaded interrupt is enabled |
| 1 | Transfer descriptor loaded interrupt is disabled |

**Bit 2 – DMAIEN** End of DMA Transfer Interrupt Enable

| Value | Description |
|---|---|
| 0 | DMA transfer completed interrupt is enabled |
| 1 | DMA transfer completed interrupt is disabled |

**Bit 1 – LFETCH** Lookup Table Fetch Enable

| Value | Description |
|---|---|
| 0 | Lookup Table DMA fetch is disabled |
| 1 | Lookup Table DMA fetch is enabled |

**Bit 0 – DFETCH** Transfer Descriptor Fetch Enable

| Value | Description |
|---|---|
| 0 | Transfer Descriptor fetch is disabled |
| 1 | Transfer Descriptor fetch is enabled |

### 38.7.26 Base DMA Next Register

**Name:** LCDC_BASENEXT
**Offset:** 0x00000068
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NEXT[31:0]** DMA Descriptor Next Address
The transfer descriptor address must be aligned on a 64-bit boundary.

### 38.7.27 Base Layer Configuration Register 0

**Name:** LCDC_BASECFG0
**Offset:** 0x0000006C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DLBO |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | BLEN[1:0] | | | | | SIF |
| Access | | | R/W | R/W | | | | R/W |
| Reset | | | 0 | 0 | | | | 0 |

**Bit 8 – DLBO** Defined Length Burst Only For Channel Bus Transaction

| Value | Description |
|---|---|
| 0 | Undefined length INCR burst is used for a burst of 2 and 3 beats. |
| 1 | Only Defined Length burst is used (SINGLE, INCR4, INCR8 and INCR16). |

**Bits 5:4 – BLEN[1:0]** AHB Burst Length

| Value | Name | Description |
|---|---|---|
| 0 | AHB_SINGLE | AHB Access is started as soon as there is enough space in the FIFO to store one data. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats. |
| 1 | AHB_INCR4 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 4 data. An AHB INCR4 Burst is used. SINGLE, INCR and INCR4 bursts are used. INCR is used for a burst of 2 and 3 beats. |
| 2 | AHB_INCR8 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 8 data. An AHB INCR8 Burst is used. SINGLE, INCR, INCR4 and INCR8 bursts are used. INCR is used for a burst of 2 and 3 beats. |

| Value | Name | Description |
|---|---|---|
| 3 | AHB_INCR16 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 16 data. An AHB INCR16 Burst is used. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats. |

**Bit 0 – SIF** Source Interface

| Value | Description |
|---|---|
| 0 | Base Layer data is retrieved through AHB interface 0. |
| 1 | Base Layer data is retrieved through AHB interface 1. |

### 38.7.28 Base Layer Configuration Register 1

**Name:** LCDC_BASECFG1
**Offset:** 0x00000070
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CLUTMODE[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RGBMODE[3:0] | | | | | | | CLUTEN |
| Access | R/W | R/W | R/W | R/W | | | | R/W |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

**Bits 9:8 – CLUTMODE[1:0]** Color Lookup Table Mode Input Selection

| Value | Name | Description |
|---|---|---|
| 0 | CLUT_1BPP | Color Lookup Table mode set to 1 bit per pixel |
| 1 | CLUT_2BPP | Color Lookup Table mode set to 2 bits per pixel |
| 2 | CLUT_4BPP | Color Lookup Table mode set to 4 bits per pixel |
| 3 | CLUT_8BPP | Color Lookup Table mode set to 8 bits per pixel |

**Bits 7:4 – RGBMODE[3:0]** RGB Mode Input Selection

| Value | Name | Description |
|---|---|---|
| 0 | 12BPP_RGB_444 | 12 bpp RGB 444 |
| 1 | 16BPP_ARGB_4444 | 16 bpp ARGB 4444 |
| 2 | 16BPP_RGBA_4444 | 16 bpp RGBA 4444 |
| 3 | 16BPP_RGB_565 | 16 bpp RGB 565 |
| 4 | 16BPP_TRGB_1555 | 16 bpp TRGB 1555 |
| 5 | 18BPP_RGB_666 | 18 bpp RGB 666 |
| 6 | 18BPP_RGB_666PACKED | 18 bpp RGB 666 PACKED |
| 7 | 19BPP_TRGB_1666 | 19 bpp TRGB 1666 |
| 8 | 19BPP_TRGB_PACKED | 19 bpp TRGB 1666 PACKED |
| 9 | 24BPP_RGB_888 | 24 bpp RGB 888 |

© 2018 Microchip Technology Inc.

| Value | Name | Description |
|---|---|---|
| 10 | 24BPP_RGB_888_PACKED | 24 bpp RGB 888 PACKED |
| 11 | 25BPP_TRGB_1888 | 25 bpp TRGB 1888 |
| 12 | 32BPP_ARGB_8888 | 32 bpp ARGB 8888 |
| 13 | 32BPP_RGBA_8888 | 32 bpp RGBA 8888 |

**Bit 0 – CLUTEN** Color Lookup Table Mode Enable

| Value | Description |
|---|---|
| 0 | RGB mode is selected. |
| 1 | Color Lookup Table mode is selected. |

### 38.7.29 Base Layer Configuration Register 2

**Name:** LCDC_BASECFG2
**Offset:** 0x00000074
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – XSTRIDE[31:0]** Horizontal Stride
XSTRIDE represents the memory offset, in bytes, between two rows of the image memory.

### 38.7.30 Base Layer Configuration Register 3

**Name:** LCDC_BASECFG3
**Offset:** 0x00000078
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RDEF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GDEF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BDEF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – RDEF[7:0]** Red Default
Default Red color when the Base DMA channel is disabled

**Bits 15:8 – GDEF[7:0]** Green Default
Default Green color when the Base DMA channel is disabled

**Bits 7:0 – BDEF[7:0]** Blue Default
Default Blue color when the Base DMA channel is disabled

### 38.7.31 Base Layer Configuration Register 4

**Name:** LCDC_BASECFG4
**Offset:** 0x0000007C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | DISCEN | | REP | DMA |
| Access | | | | | R/W | | R/W | R/W |
| Reset | | | | | 0 | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 11 – DISCEN** Discard Area Enable

| Value | Description |
|---|---|
| 0 | The whole frame is retrieved from memory. |
| 1 | The DMA channel discards the area located at screen coordinate {DISCXPOS, DISCYPOS}. |

**Bit 9 – REP** Use Replication logic to expand RGB color to 24 bits

| Value | Description |
|---|---|
| 0 | When the selected pixel depth is less than 24 bpp the pixel is shifted and least significant bits are set to 0. |
| 1 | When the selected pixel depth is less than 24 bpp the pixel is shifted and the least significant bit replicates the msb. |

**Bit 8 – DMA** Use DMA Data Path

| Value | Description |
|---|---|
| 0 | The default color is used on the Base Layer. |
| 1 | The DMA channel retrieves the pixels stream from the memory. |

### 38.7.32 Base Layer Configuration Register 5

**Name:** LCDC_BASECFG5
**Offset:** 0x00000080
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | DISCYPOS[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DISCYPOS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | DISCXPOS[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DISCXPOS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 26:16 – DISCYPOS[10:0]** Discard Area Vertical Coordinate
Vertical Position of the Discard Area

**Bits 10:0 – DISCXPOS[10:0]** Discard Area Horizontal Coordinate
Horizontal Position of the Discard Area

### 38.7.33 Base Layer Configuration Register 6

**Name:** LCDC_BASECFG6
**Offset:** 0x00000084
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | DISCYSIZE[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DISCYSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | DISCXSIZE[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DISCXSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 26:16 – DISCYSIZE[10:0]** Discard Area Vertical Size
Discard Vertical size in pixels. The Discard size is set to (DISCYSIZE + 1) pixels vertically.

**Bits 10:0 – DISCXSIZE[10:0]** Discard Area Horizontal Size
Discard Horizontal size in pixels. The Discard size is set to (DISCXSIZE + 1) pixels horizontally.

### 38.7.34 Overlay 1 Channel Enable Register

**Name:** LCDC_OVR1CHER
**Offset:** 0x00000140
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | A2QEN | UPDATEEN | CHEN |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

**Bit 2 – A2QEN** Add To Queue Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Indicates that a valid descriptor has been written to memory, its memory location should be written to the DMA head pointer. The A2QSR status bit is set to one, and it is reset by hardware as soon as the descriptor pointed to by the DMA head pointer is added to the list. |

**Bit 1 – UPDATEEN** Update Overlay Attributes Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Updates window attributes (size, alpha blending, etc.) on the next start of frame. |

**Bit 0 – CHEN** Channel Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Enables the DMA channel |

### 38.7.35 Overlay 1 Channel Disable Register

**Name:** LCDC_OVR1CHDR
**Offset:** 0x00000144
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 8 – CHRST** Channel Reset

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Resets the layer immediately. The frame is aborted. |

**Bit 0 – CHDIS** Channel Disable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Disables the layer at the end of the current frame. The frame is completed. |

### 38.7.36 Overlay 1 Channel Status Register

**Name:** LCDC_OVR1CHSR
**Offset:** 0x00000148
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | A2QSR | UPDATESR | CHSR |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – A2QSR** Add To Queue Status

| Value | Description |
|---|---|
| 0 | Add to queue not pending |
| 1 | Add to queue pending |

**Bit 1 – UPDATESR** Update Overlay Attributes In Progress Status

| Value | Description |
|---|---|
| 0 | No update pending |
| 1 | Overlay attributes will be updated on the next frame |

**Bit 0 – CHSR** Channel Status

| Value | Description |
|---|---|
| 0 | Layer disabled |
| 1 | Layer enabled |

### 38.7.37 Overlay 1 Interrupt Enable Register

**Name:** LCDC_OVR1IER
**Offset:** 0x0000014C
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | OVR | DONE | ADD | DSCR | DMA | | |
| Access | | W | W | W | W | W | | |
| Reset | | – | – | – | – | – | | |

**Bit 6 – OVR** Overflow Interrupt Enable

**Bit 5 – DONE** End of List Interrupt Enable

**Bit 4 – ADD** Head Descriptor Loaded Interrupt Enable

**Bit 3 – DSCR** Descriptor Loaded Interrupt Enable

**Bit 2 – DMA** End of DMA Transfer Interrupt Enable

### 38.7.38 Overlay 1 Interrupt Disable Register

**Name:** LCDC_OVR1IDR
**Offset:** 0x00000150
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | OVR | DONE | ADD | DSCR | DMA | | |
| Access | | W | W | W | W | W | | |
| Reset | | – | – | – | – | – | | |

**Bit 6 – OVR** Overflow Interrupt Disable

**Bit 5 – DONE** End of List Interrupt Disable

**Bit 4 – ADD** Head Descriptor Loaded Interrupt Disable

**Bit 3 – DSCR** Descriptor Loaded Interrupt Disable

**Bit 2 – DMA** End of DMA Transfer Interrupt Disable

### 38.7.39 Overlay 1 Interrupt Mask Register

**Name:** LCDC_OVR1IMR
**Offset:** 0x00000154
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:
0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | OVR | DONE | ADD | DSCR | DMA | | |
| Access | | R | R | R | R | R | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | | |

**Bit 6 – OVR** Overflow Interrupt Mask

**Bit 5 – DONE** End of List Interrupt Mask

**Bit 4 – ADD** Head Descriptor Loaded Interrupt Mask

**Bit 3 – DSCR** Descriptor Loaded Interrupt Mask

**Bit 2 – DMA** End of DMA Transfer Interrupt Mask

### 38.7.40 Overlay 1 Interrupt Status Register

**Name:** LCDC_OVR1ISR
**Offset:** 0x00000158
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | OVR | DONE | ADD | DSCR | DMA | | |
| Access | | R | R | R | R | R | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | | |

**Bit 6 – OVR** Overflow Detected

| Value | Description |
|---|---|
| 0 | No overflow occurred since last read of LCDC_OVR1ISR |
| 1 | An overflow occurred. This flag is reset after a read operation. |

**Bit 5 – DONE** End of List Detected

| Value | Description |
|---|---|
| 0 | No End of List condition has occurred since last read of LCDC_OVR1ISR |
| 1 | End of List condition has occurred. This flag is reset after a read operation. |

**Bit 4 – ADD** Head Descriptor Loaded

| Value | Description |
|---|---|
| 0 | No descriptor has been loaded since last read of LCDC_OVR1ISR |
| 1 | The descriptor pointed to by the LCDC_OVR1HEAD register has been loaded successfully. This flag is reset after a read operation. |

**Bit 3 – DSCR** DMA Descriptor Loaded

| Value | Description |
|---|---|
| 0 | No descriptor has been loaded since last read of LCDC_OVR1ISR |

| Value | Description |
|---|---|
| 1 | A descriptor has been loaded successfully. This flag is reset after a read operation. |

**Bit 2 – DMA** End of DMA Transfer

| Value | Description |
|---|---|
| 0 | No End of Transfer has been detected since last read of LCDC_OVR1ISR |
| 1 | End of Transfer has been detected. This flag is reset after a read operation. |

### 38.7.41 Overlay 1 Head Register

**Name:** LCDC_OVR1HEAD
**Offset:** 0x0000015C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – HEAD[29:0]** DMA Head Pointer
The Head Pointer points to a new descriptor.

### 38.7.42 Overlay 1 Address Register

**Name:** LCDC_OVR1ADDR
**Offset:** 0x00000160
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** DMA Transfer Overlay 1 Address
Overlay 1 frame buffer base address

### 38.7.43 Overlay 1 Control Register

**Name:** LCDC_OVR1CTRL
**Offset:** 0x00000164
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DONEIEN | ADDIEN | DSCRIEN | DMAIEN | LFETCH | DFETCH |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 5 – DONEIEN** End of List Interrupt Enable

| Value | Description |
|---|---|
| 0 | End of list interrupt is disabled |
| 1 | End of list interrupt is enabled |

**Bit 4 – ADDIEN** Add Head Descriptor to Queue Interrupt Enable

| Value | Description |
|---|---|
| 0 | Transfer descriptor added to queue interrupt is enabled |
| 1 | Transfer descriptor added to queue interrupt is enabled |

**Bit 3 – DSCRIEN** Descriptor Loaded Interrupt Enable

| Value | Description |
|---|---|
| 0 | Transfer descriptor loaded interrupt is enabled |
| 1 | Transfer descriptor loaded interrupt is disabled |

**Bit 2 – DMAIEN** End of DMA Transfer Interrupt Enable

| Value | Description |
|---|---|
| 0 | DMA transfer completed interrupt is enabled |
| 1 | DMA transfer completed interrupt is disabled |

**Bit 1 – LFETCH** Lookup Table Fetch Enable

| Value | Description |
|---|---|
| 0 | Lookup Table DMA fetch is disabled |
| 1 | Lookup Table DMA fetch is enabled |

**Bit 0 – DFETCH** Transfer Descriptor Fetch Enable

| Value | Description |
|---|---|
| 0 | Transfer Descriptor fetch is disabled |
| 1 | Transfer Descriptor fetch is enabled |

### 38.7.44 Overlay 1 Next Register

**Name:** LCDC_OVR1NEXT
**Offset:** 0x00000168
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NEXT[31:0]** DMA Descriptor Next Address
The transfer descriptor address must be aligned on a 64-bit boundary.

### 38.7.45 Overlay 1 Configuration Register 0

**Name:** LCDC_OVR1CFG0
**Offset:** 0x00000170
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | LOCKDIS | ROTDIS | | | | DLBO |
| Access | | | R/W | R/W | | | | R/W |
| Reset | | | 0 | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | BLEN[1:0] | | | | | SIF |
| Access | | | R/W | R/W | | | | R/W |
| Reset | | | 0 | 0 | | | | 0 |

**Bit 13 – LOCKDIS** Hardware Rotation Lock Disable

| Value | Description |
|---|---|
| 0 | AHB lock signal is asserted when a rotation is performed. |
| 1 | AHB lock signal is cleared when a rotation is performed. |

**Bit 12 – ROTDIS** Hardware Rotation Optimization Disable

| Value | Description |
|---|---|
| 0 | Rotation optimization is enabled. |
| 1 | Rotation optimization is disabled. |

**Bit 8 – DLBO** Defined Length Burst Only for Channel Bus Transaction

| Value | Description |
|---|---|
| 0 | Undefined length INCR burst is used for a burst of 2 and 3 beats. |
| 1 | Only Defined Length burst is used (SINGLE, INCR4, INCR8 and INCR16). |

**Bits 5:4 – BLEN[1:0]** AHB Burst Length

| Value | Name | Description |
|---|---|---|
| 0 | AHB_BLEN_SINGLE | AHB Access is started as soon as there is enough space in the FIFO to store one data. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats. |
| 1 | AHB_BLEN_INCR4 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 4 data. An AHB INCR4 Burst is used. SINGLE, INCR and INCR4 bursts are used. INCR is used for a burst of 2 and 3 beats. |
| 2 | AHB_BLEN_INCR8 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 8 data. An AHB INCR8 Burst is used. SINGLE, INCR, INCR4 and INCR8 bursts are used. INCR is used for a burst of 2 and 3 beats. |
| 3 | AHB_BLEN_INCR16 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 16 data. An AHB INCR16 Burst is used. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats. |

**Bit 0 – SIF** Source Interface

| Value | Description |
|---|---|
| 0 | Base Layer data is retrieved through AHB interface 0. |
| 1 | Base Layer data is retrieved through AHB interface 1. |

### 38.7.46 Overlay 1 Configuration Register 1

**Name:** LCDC_OVR1CFG1
**Offset:** 0x00000170
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CLUTMODE[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RGBMODE[3:0] | | | | | | | CLUTEN |
| Access | R/W | R/W | R/W | R/W | | | | R/W |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

**Bits 9:8 – CLUTMODE[1:0]** Color Lookup Table Mode Input Selection

| Value | Name | Description |
|---|---|---|
| 0 | CLUT_1BPP | Color Lookup Table mode set to 1 bit per pixel |
| 1 | CLUT_2BPP | Color Lookup Table mode set to 2 bits per pixel |
| 2 | CLUT_4BPP | Color Lookup Table mode set to 4 bits per pixel |
| 3 | CLUT_8BPP | Color Lookup Table mode set to 8 bits per pixel |

**Bits 7:4 – RGBMODE[3:0]** RGB Mode Input Selection

| Value | Name | Description |
|---|---|---|
| 0 | 12BPP_RGB_444 | 12 bpp RGB 444 |
| 1 | 16BPP_ARGB_4444 | 16 bpp ARGB 4444 |
| 2 | 16BPP_RGBA_4444 | 16 bpp RGBA 4444 |
| 3 | 16BPP_RGB_565 | 16 bpp RGB 565 |
| 4 | 16BPP_TRGB_1555 | 16 bpp TRGB 1555 |
| 5 | 18BPP_RGB_666 | 18 bpp RGB 666 |
| 6 | 18BPP_RGB_666PACKED | 18 bpp RGB 666 PACKED |
| 7 | 19BPP_TRGB_1666 | 19 bpp TRGB 1666 |
| 8 | 19BPP_TRGB_PACKED | 19 bpp TRGB 1666 PACKED |
| 9 | 24BPP_RGB_888 | 24 bpp RGB 888 |

| Value | Name | Description |
|---|---|---|
| 10 | 24BPP_RGB_888_PACKED | 24 bpp RGB 888 PACKED |
| 11 | 25BPP_TRGB_1888 | 25 bpp TRGB 1888 |
| 12 | 32BPP_ARGB_8888 | 32 bpp ARGB 8888 |
| 13 | 32BPP_RGBA_8888 | 32 bpp RGBA 8888 |

**Bit 0 – CLUTEN** Color Lookup Table Mode Enable

| Value | Description |
|---|---|
| 0 | RGB mode is selected. |
| 1 | Color Lookup Table mode is selected. |

### 38.7.47 Overlay 1 Configuration Register 2

**Name:** LCDC_OVR1CFG2
**Offset:** 0x00000174
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | YPOS[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YPOS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | XPOS[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPOS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 26:16 – YPOS[10:0]** Vertical Window Position
Overlay 1 Vertical window position.

**Bits 10:0 – XPOS[10:0]** Horizontal Window Position
Overlay 1 Horizontal window position.

### 38.7.48 Overlay 1 Configuration Register 3

**Name:** LCDC_OVR1CFG3
**Offset:** 0x00000178
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | YSIZE[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | XSIZE[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 26:16 – YSIZE[10:0]** Vertical Window Size
Overlay 1 window height in pixels. The window height is set to (YSIZE + 1).

The following constraint must be met: YPOS + YSIZE ≤ RPF

**Bits 10:0 – XSIZE[10:0]** Horizontal Window Size
Overlay 1 window width in pixels. The window width is set to (XSIZE + 1).

The following constraint must be met: XPOS + XSIZE ≤ PPL

### 38.7.49 Overlay 1 Configuration Register 4

**Name:** LCDC_OVR1CFG4
**Offset:** 0x0000017C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – XSTRIDE[31:0]** Horizontal Stride
XSTRIDE represents the memory offset, in bytes, between two rows of the image memory.

### 38.7.50 Overlay 1 Configuration Register 5

**Name:** LCDC_OVR1CFG5
**Offset:** 0x00000180
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PSTRIDE[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PSTRIDE[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PSTRIDE[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PSTRIDE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – PSTRIDE[31:0]** Pixel Stride
PSTRIDE represents the memory offset, in bytes, between two pixels of the image.

### 38.7.51 Overlay 1 Configuration Register 6

**Name:** LCDC_OVR1CFG6
**Offset:** 0x00000184
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RDEF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GDEF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BDEF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – RDEF[7:0]** Red Default
Default Red color when the Overlay 1 DMA channel is disabled.

**Bits 15:8 – GDEF[7:0]** Green Default
Default Green color when the Overlay 1 DMA channel is disabled.

**Bits 7:0 – BDEF[7:0]** Blue Default
Default Blue color when the Overlay 1 DMA channel is disabled.

### 38.7.52 Overlay 1 Configuration Register 7

**Name:** LCDC_OVR1CFG7
**Offset:** 0x00000188
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RKEY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GKEY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BKEY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – RKEY[7:0]** Red Color Component Chroma Key
Reference Red chroma key used to match the Red color of the current overlay.

**Bits 15:8 – GKEY[7:0]** Green Color Component Chroma Key
Reference Green chroma key used to match the Green color of the current overlay.

**Bits 7:0 – BKEY[7:0]** Blue Color Component Chroma Key
Reference Blue chroma key used to match the Blue color of the current overlay.

### 38.7.53 Overlay 1 Configuration Register 8

**Name:** LCDC_OVR1CFG8
**Offset:** 0x0000018C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RMASK[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GMASK[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BMASK[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – RMASK[7:0]** Red Color Component Chroma Key Mask
Red Mask used when the compare function is used. If a bit is set then this bit is compared.

**Bits 15:8 – GMASK[7:0]** Green Color Component Chroma Key Mask
Green Mask used when the compare function is used. If a bit is set then this bit is compared.

**Bits 7:0 – BMASK[7:0]** Blue Color Component Chroma Key Mask
Blue Mask used when the compare function is used. If a bit is set then this bit is compared.

### 38.7.54 Overlay 1 Configuration Register 9

**Name:** LCDC_OVR1CFG9
**Offset:** 0x00000190
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | DSTKEY | REP | DMA |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVR | LAEN | GAEN | REVALPHA | ITER | ITER2BL | INV | CRKEY |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – GA[7:0]** Blender Global Alpha
Global alpha blender for the current layer.

**Bit 10 – DSTKEY** Destination Chroma Keying

| Value | Description |
|---|---|
| 0 | Source Chroma keying is enabled. |
| 1 | Destination Chroma keying is used. |

**Bit 9 – REP** Use Replication logic to expand RGB color to 24 bits

| Value | Description |
|---|---|
| 0 | When the selected pixel depth is less than 24 bpp the pixel is shifted and least significant bits are set to 0. |
| 1 | When the selected pixel depth is less than 24 bpp the pixel is shifted and the least significant bit replicates the msb. |

**Bit 8 – DMA** Blender DMA Layer Enable

| Value | Description |
|---|---|
| 0 | The default color is used on the Overlay 1 Layer. |
| 1 | The DMA channel retrieves the pixels stream from the memory. |

**Bit 7 – OVR** Blender Overlay Layer Enable

| Value | Description |
|---|---|
| 0 | Overlay pixel color is set to the default overlay pixel color. |
| 1 | Overlay pixel color is set to the DMA channel pixel color. |

**Bit 6 – LAEN** Blender Local Alpha Enable

| Value | Description |
|---|---|
| 0 | Local alpha blending coefficient is disabled. |
| 1 | Local alpha blending coefficient is enabled. |

**Bit 5 – GAEN** Blender Global Alpha Enable

| Value | Description |
|---|---|
| 0 | Global alpha blending coefficient is disabled. |
| 1 | Global alpha blending coefficient is enabled. |

**Bit 4 – REVALPHA** Blender Reverse Alpha

| Value | Description |
|---|---|
| 0 | Pixel difference is multiplied by alpha. |
| 1 | Pixel difference is multiplied by 1 - alpha. |

**Bit 3 – ITER** Blender Use Iterated Color

| Value | Description |
|---|---|
| 0 | Pixel difference is set to 0. |
| 1 | Pixel difference is set to the iterated pixel value. |

**Bit 2 – ITER2BL** Blender Iterated Color Enable

| Value | Description |
|---|---|
| 0 | Final adder stage operand is set to 0. |
| 1 | Final adder stage operand is set to the iterated pixel value. |

**Bit 1 – INV** Blender Inverted Blender Output Enable

| Value | Description |
|---|---|
| 0 | Iterated pixel is the blended pixel. |
| 1 | Iterated pixel is the inverted pixel. |

**Bit 0 – CRKEY** Blender Chroma Key Enable

| Value | Description |
|---|---|
| 0 | Chroma key matching is disabled. |
| 1 | Chroma key matching is enabled. |

### 38.7.55 Overlay 2 Channel Enable Register

**Name:** LCDC_OVR2CHER
**Offset:** 0x00000240
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | A2QEN | UPDATEEN | CHEN |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

**Bit 2 – A2QEN** Add To Queue Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Indicates that a valid descriptor has been written to memory, its memory location should be written to the DMA head pointer. The A2QSR status bit is set to one, and it is reset by hardware as soon as the descriptor pointed to by the DMA head pointer is added to the list. |

**Bit 1 – UPDATEEN** Update Overlay Attributes Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Updates windows attributes on the next start of frame. |

**Bit 0 – CHEN** Channel Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Enables the DMA channel |

### 38.7.56 Overlay 2 Channel Disable Register

**Name:** LCDC_OVR2CHDR
**Offset:** 0x00000244
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 8 – CHRST** Channel Reset

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Resets the layer immediately. The frame is aborted. |

**Bit 0 – CHDIS** Channel Disable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Disables the layer at the end of the current frame. The frame is completed. |

### 38.7.57 Overlay 2 Channel Status Register

**Name:** LCDC_OVR2CHSR
**Offset:** 0x00000248
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | A2QSR | UPDATESR | CHSR |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – A2QSR** Add To Queue Status

| Value | Description |
|---|---|
| 0 | Add to queue not pending |
| 1 | Add to queue pending |

**Bit 1 – UPDATESR** Update Overlay Attributes In Progress Status

| Value | Description |
|---|---|
| 0 | No update pending |
| 1 | Overlay attributes will be updated on the next frame |

**Bit 0 – CHSR** Channel Status

| Value | Description |
|---|---|
| 0 | Layer disabled |
| 1 | Layer enabled |

### 38.7.58 Overlay 2 Interrupt Enable Register

**Name:** LCDC_OVR2IER
**Offset:** 0x0000024C
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | OVR | DONE | ADD | DSCR | DMA | | |
| Access | | W | W | W | W | W | | |
| Reset | | – | – | – | – | – | | |

**Bit 6 – OVR** Overflow Interrupt Enable

**Bit 5 – DONE** End of List Interrupt Enable

**Bit 4 – ADD** Head Descriptor Loaded Interrupt Enable

**Bit 3 – DSCR** Descriptor Loaded Interrupt Enable

**Bit 2 – DMA** End of DMA Transfer Interrupt Enable

### 38.7.59 Overlay 2 Interrupt Disable Register

**Name:** LCDC_OVR2IDR
**Offset:** 0x00000250
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | OVR | DONE | ADD | DSCR | DMA | | |
| Access | | W | W | W | W | W | | |
| Reset | | – | – | – | – | – | | |

**Bit 6 – OVR** Overflow Interrupt Disable

**Bit 5 – DONE** End of List Interrupt Disable

**Bit 4 – ADD** Head Descriptor Loaded Interrupt Disable

**Bit 3 – DSCR** Descriptor Loaded Interrupt Disable

**Bit 2 – DMA** End of DMA Transfer Interrupt Disable

### 38.7.60 Overlay 2 Interrupt Mask Register

**Name:** LCDC_OVR2IMR
**Offset:** 0x00000254
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:
0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | OVR | DONE | ADD | DSCR | DMA | | |
| Access | | R | R | R | R | R | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | | |

**Bit 6 – OVR** Overflow Interrupt Mask

**Bit 5 – DONE** End of List Interrupt Mask

**Bit 4 – ADD** Head Descriptor Loaded Interrupt Mask

**Bit 3 – DSCR** Descriptor Loaded Interrupt Mask

**Bit 2 – DMA** End of DMA Transfer Interrupt Mask

### 38.7.61 Overlay 2 Interrupt Status Register

**Name:** LCDC_OVR2ISR
**Offset:** 0x00000258
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | OVR | DONE | ADD | DSCR | DMA | | |
| Access | | R | R | R | R | R | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | | |

**Bit 6 – OVR** Overflow Detected

| Value | Description |
|---|---|
| 0 | No overflow occurred since last read of LCDC_OVR2ISR |
| 1 | An overflow occurred. This flag is reset after a read operation. |

**Bit 5 – DONE** End of List Detected

| Value | Description |
|---|---|
| 0 | No End of List condition occurred since last read of LCDC_OVR2ISR |
| 1 | End of List condition has occurred. This flag is reset after a read operation. |

**Bit 4 – ADD** Head Descriptor Loaded

| Value | Description |
|---|---|
| 0 | No descriptor has been loaded since last read of LCDC_OVR2ISR |
| 1 | The descriptor pointed to by the LCDC_OVR2HEAD register has been loaded successfully. This flag is reset after a read operation. |

**Bit 3 – DSCR** DMA Descriptor Loaded

| Value | Description |
|---|---|
| 0 | No descriptor has been loaded since last read of LCDC_OVR2ISR |

| Value | Description |
|---|---|
| 1 | A descriptor has been loaded successfully. This flag is reset after a read operation. |

**Bit 2 – DMA** End of DMA Transfer

| Value | Description |
|---|---|
| 0 | No End of Transfer has been detected since last read of LCDC_OVR2ISR |
| 1 | End of Transfer has been detected. This flag is reset after a read operation. |

### 38.7.62 Overlay 2 Head Register

**Name:** LCDC_OVR2HEAD
**Offset:** 0x0000025C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – HEAD[29:0]** DMA Head Pointer
The Head Pointer points to a new descriptor.

### 38.7.63 Overlay 2 Address Register

**Name:** LCDC_OVR2ADDR
**Offset:** 0x00000260
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** DMA Transfer Overlay 2 Address
Overlay 2 frame buffer base address.

### 38.7.64 Overlay 2 Control Register

**Name:** LCDC_OVR2CTRL
**Offset:** 0x00000264
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DONEIEN | ADDIEN | DSCRIEN | DMAIEN | LFETCH | DFETCH |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 5 – DONEIEN** End of List Interrupt Enable

| Value | Description |
|---|---|
| 0 | End of list interrupt is disabled. |
| 1 | End of list interrupt is enabled. |

**Bit 4 – ADDIEN** Add Head Descriptor to Queue Interrupt Enable

| Value | Description |
|---|---|
| 0 | Transfer descriptor added to queue interrupt is enabled. |
| 1 | Transfer descriptor added to queue interrupt is enabled. |

**Bit 3 – DSCRIEN** Descriptor Loaded Interrupt Enable

| Value | Description |
|---|---|
| 0 | Transfer descriptor loaded interrupt is enabled. |
| 1 | Transfer descriptor loaded interrupt is disabled. |

**Bit 2 – DMAIEN** End of DMA Transfer Interrupt Enable

| Value | Description |
|---|---|
| 0 | DMA transfer completed interrupt is enabled. |
| 1 | DMA transfer completed interrupt is disabled. |

**Bit 1 – LFETCH** Lookup Table Fetch Enable

| Value | Description |
|---|---|
| 0 | Lookup Table DMA fetch is disabled. |
| 1 | Lookup Table DMA fetch is enabled. |

**Bit 0 – DFETCH** Transfer Descriptor Fetch Enable

| Value | Description |
|---|---|
| 0 | Transfer Descriptor fetch is disabled. |
| 1 | Transfer Descriptor fetch is enabled. |

### 38.7.65 Overlay 2 Next Register

**Name:** LCDC_OVR2NEXT
**Offset:** 0x00000268
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NEXT[31:0]** DMA Descriptor Next Address
The transfer descriptor address must be aligned on a 64-bit boundary.

### 38.7.66 Overlay 2 Configuration Register 0

**Name:** LCDC_OVR2CFG0
**Offset:** 0x0000026C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | LOCKDIS | ROTDIS | | | | DLBO |
| Access | | | R/W | R/W | | | | R/W |
| Reset | | | 0 | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | BLEN[1:0] | | | | | |
| Access | | | R/W | R/W | | | | |
| Reset | | | 0 | 0 | | | | |

**Bit 13 – LOCKDIS** Hardware Rotation Lock Disable

| Value | Description |
|---|---|
| 0 | AHB lock signal is asserted when a rotation is performed. |
| 1 | AHB lock signal is cleared when a rotation is performed. |

**Bit 12 – ROTDIS** Hardware Rotation Optimization Disable

| Value | Description |
|---|---|
| 0 | Rotation optimization is enabled. |
| 1 | Rotation optimization is disabled. |

**Bit 8 – DLBO** Defined Length Burst Only For Channel Bus Transaction

| Value | Description |
|---|---|
| 0 | Undefined length INCR burst is used for 2 and 3 beats burst. |
| 1 | Only Defined Length burst is used (SINGLE, INCR4, INCR8 and INCR16). |

**Bits 5:4 – BLEN[1:0]** AHB Burst Length

| Value | Name | Description |
|---|---|---|
| 0 | AHB_SINGLE | AHB Access is started as soon as there is enough space in the FIFO to store one data. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats. |
| 1 | AHB_INCR4 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 4 data. An AHB INCR4 Burst is used. SINGLE, INCR and INCR4 bursts are used. INCR is used for a burst of 2 and 3 beats. |
| 2 | AHB_INCR8 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 8 data. An AHB INCR8 Burst is used. SINGLE, INCR, INCR4 and INCR8 bursts are used. INCR is used for a burst of 2 and 3 beats. |
| 3 | AHB_INCR16 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 16 data. An AHB INCR16 Burst is used. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats. |

### 38.7.67 Overlay 2 Configuration Register 1

**Name:** LCDC_OVR2CFG1
**Offset:** 0x00000270
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CLUTMODE[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RGBMODE[3:0] | | | | | | | CLUTEN |
| Access | R/W | R/W | R/W | R/W | | | | R/W |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

**Bits 9:8 – CLUTMODE[1:0]** Color Lookup Table Mode Input Selection

| Value | Name | Description |
|---|---|---|
| 0 | CLUT_1BPP | Color Lookup Table mode set to 1 bit per pixel |
| 1 | CLUT_2BPP | Color Lookup Table mode set to 2 bits per pixel |
| 2 | CLUT_4BPP | Color Lookup Table mode set to 4 bits per pixel |
| 3 | CLUT_8BPP | Color Lookup Table mode set to 8 bits per pixel |

**Bits 7:4 – RGBMODE[3:0]** RGB Mode Input Selection

| Value | Name | Description |
|---|---|---|
| 0 | 12BPP_RGB_444 | 12 bpp RGB 444 |
| 1 | 16BPP_ARGB_4444 | 16 bpp ARGB 4444 |
| 2 | 16BPP_RGBA_4444 | 16 bpp RGBA 4444 |
| 3 | 16BPP_RGB_565 | 16 bpp RGB 565 |
| 4 | 16BPP_TRGB_1555 | 16 bpp TRGB 1555 |
| 5 | 18BPP_RGB_666 | 18 bpp RGB 666 |
| 6 | 18BPP_RGB_666PACKED | 18 bpp RGB 666 PACKED |
| 7 | 19BPP_TRGB_1666 | 19 bpp TRGB 1666 |
| 8 | 19BPP_TRGB_PACKED | 19 bpp TRGB 1666 PACKED |
| 9 | 24BPP_RGB_888 | 24 bpp RGB 888 |

| Value | Name | Description |
|---|---|---|
| 10 | 24BPP_RGB_888_PACKED | 24 bpp RGB 888 PACKED |
| 11 | 25BPP_TRGB_1888 | 25 bpp TRGB 1888 |
| 12 | 32BPP_ARGB_8888 | 32 bpp ARGB 8888 |
| 13 | 32BPP_RGBA_8888 | 32 bpp RGBA 8888 |

**Bit 0 – CLUTEN** Color Lookup Table Mode Enable

| Value | Description |
|---|---|
| 0 | RGB mode is selected. |
| 1 | Color Lookup Table mode is selected. |

### 38.7.68 Overlay 2 Configuration Register 2

**Name:** LCDC_OVR2CFG2
**Offset:** 0x00000274
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | YPOS[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YPOS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | XPOS[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPOS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 26:16 – YPOS[10:0]** Vertical Window Position
Overlay 2 Vertical window position.

**Bits 10:0 – XPOS[10:0]** Horizontal Window Position
Overlay 2 Horizontal window position.

### 38.7.69 Overlay 2 Configuration Register 3

**Name:** LCDC_OVR2CFG3
**Offset:** 0x00000278
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | YSIZE[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | XSIZE[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 26:16 – YSIZE[10:0]** Vertical Window Size
Overlay 2 window height in pixels. The window height is set to (YSIZE + 1).

The following constraint must be met: YPOS + YSIZE ≤ RPF

**Bits 10:0 – XSIZE[10:0]** Horizontal Window Size
Overlay 2 window width in pixels. The window width is set to (XSIZE + 1).

The following constraint must be met: XPOS + XSIZE ≤ PPL

### 38.7.70 Overlay 2 Configuration Register 4

**Name:** LCDC_OVR2CFG4
**Offset:** 0x0000027C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – XSTRIDE[31:0]** Horizontal Stride
XSTRIDE represents the memory offset, in bytes, between two rows of the image memory.

### 38.7.71 Overlay 2 Configuration Register 5

**Name:** LCDC_OVR2CFG5
**Offset:** 0x00000280
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PSTRIDE[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PSTRIDE[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PSTRIDE[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PSTRIDE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – PSTRIDE[31:0]** Pixel Stride
PSTRIDE represents the memory offset, in bytes, between two pixels of the image memory.

### 38.7.72 Overlay 2 Configuration Register 6

**Name:** LCDC_OVR2CFG6
**Offset:** 0x00000284
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RDEF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GDEF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BDEF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – RDEF[7:0]** Red Default
Default Red color when the Overlay 1 DMA channel is disabled.

**Bits 15:8 – GDEF[7:0]** Green Default
Default Green color when the Overlay 1 DMA channel is disabled.

**Bits 7:0 – BDEF[7:0]** Blue Default
Default Blue color when the Overlay 1 DMA channel is disabled.

### 38.7.73 Overlay 2 Configuration Register 7

**Name:** LCDC_OVR2CFG7
**Offset:** 0x00000288
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RKEY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GKEY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BKEY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – RKEY[7:0]** Red Color Component Chroma Key
Reference Red chroma key used to match the Red color of the current overlay.

**Bits 15:8 – GKEY[7:0]** Green Color Component Chroma Key
Reference Green chroma key used to match the Green color of the current overlay.

**Bits 7:0 – BKEY[7:0]** Blue Color Component Chroma Key
Reference Blue chroma key used to match the Blue color of the current overlay.

### 38.7.74 Overlay 2 Configuration Register 8

**Name:** LCDC_OVR2CFG8
**Offset:** 0x0000028C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RMASK[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GMASK[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BMASK[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – RMASK[7:0]** Red Color Component Chroma Key Mask
Red Mask used when the compare function is used. If a bit is set then this bit is compared.

**Bits 15:8 – GMASK[7:0]** Green Color Component Chroma Key Mask
Green Mask used when the compare function is used. If a bit is set then this bit is compared.

**Bits 7:0 – BMASK[7:0]** Blue Color Component Chroma Key Mask
Blue Mask used when the compare function is used. If a bit is set then this bit is compared.

### 38.7.75 Overlay 2 Configuration Register 9

**Name:** LCDC_OVR2CFG9
**Offset:** 0x00000290
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | DSTKEY | REP | DMA |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVR | LAEN | GAEN | REVALPHA | ITER | ITER2BL | INV | CRKEY |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – GA[7:0]** Blender Global Alpha
Global alpha blender for the current layer.

**Bit 10 – DSTKEY** Destination Chroma Keying

| Value | Description |
|---|---|
| 0 | Source Chroma keying is enabled. |
| 1 | Destination Chroma keying is used. |

**Bit 9 – REP** Use Replication logic to expand RGB color to 24 bits

| Value | Description |
|---|---|
| 0 | When the selected pixel depth is less than 24 bpp the pixel is shifted and least significant bits are set to 0. |
| 1 | When the selected pixel depth is less than 24 bpp the pixel is shifted and the least significant bit replicates the msb. |

**Bit 8 – DMA** Blender DMA Layer Enable

| Value | Description |
|---|---|
| 0 | The default color is used on the Overlay 1 Layer. |
| 1 | The DMA channel retrieves the pixels stream from the memory. |

**Bit 7 – OVR** Blender Overlay Layer Enable

| Value | Description |
|---|---|
| 0 | Overlay pixel color is set to the default overlay pixel color. |
| 1 | Overlay pixel color is set to the DMA channel pixel color. |

**Bit 6 – LAEN** Blender Local Alpha Enable

| Value | Description |
|---|---|
| 0 | Local alpha blending coefficient is disabled. |
| 1 | Local alpha blending coefficient is enabled. |

**Bit 5 – GAEN** Blender Global Alpha Enable

| Value | Description |
|---|---|
| 0 | Global alpha blending coefficient is disabled. |
| 1 | Global alpha blending coefficient is enabled. |

**Bit 4 – REVALPHA** Blender Reverse Alpha

| Value | Description |
|---|---|
| 0 | Pixel difference is multiplied by alpha. |
| 1 | Pixel difference is multiplied by 1 - alpha. |

**Bit 3 – ITER** Blender Use Iterated Color

| Value | Description |
|---|---|
| 0 | Pixel difference is set to 0. |
| 1 | Pixel difference is set to the iterated pixel value. |

**Bit 2 – ITER2BL** Blender Iterated Color Enable

| Value | Description |
|---|---|
| 0 | Final adder stage operand is set to 0. |
| 1 | Final adder stage operand is set to the iterated pixel value. |

**Bit 1 – INV** Blender Inverted Blender Output Enable

| Value | Description |
|---|---|
| 0 | Iterated pixel is the blended pixel. |
| 1 | Iterated pixel is the inverted pixel. |

**Bit 0 – CRKEY** Blender Chroma Key Enable

| Value | Description |
|---|---|
| 0 | Chroma key matching is disabled. |
| 1 | Chroma key matching is enabled. |

### 38.7.76 High-End Overlay Channel Enable Register

**Name:** LCDC_HEOCHER
**Offset:** 0x00000340
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | A2QEN | UPDATEEN | CHEN |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

**Bit 2 – A2QEN** Add To Queue Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Indicates that a valid descriptor has been written to memory, its memory location should be written to the DMA head pointer. The A2QSR status bit is set to one, and it is reset by hardware as soon as the descriptor pointed to by the DMA head pointer is added to the list. |

**Bit 1 – UPDATEEN** Update Overlay Attributes Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Updates windows attributes on the next start of frame. |

**Bit 0 – CHEN** Channel Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Enables the DMA channel |

### 38.7.77 High-End Overlay Channel Disable Register

**Name:** LCDC_HEOCHDR
**Offset:** 0x00000344
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 8 – CHRST** Channel Reset

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Resets the layer immediately. The frame is aborted. |

**Bit 0 – CHDIS** Channel Disable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Disables the layer at the end of the current frame. The frame is completed. |

### 38.7.78 High-End Overlay Channel Status Register

**Name:** LCDC_HEOCHSR
**Offset:** 0x00000348
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | A2QSR | UPDATESR | CHSR |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – A2QSR** Add To Queue Status

| Value | Description |
|---|---|
| 0 | Add to queue not pending |
| 1 | Add to queue pending |

**Bit 1 – UPDATESR** Update Overlay Attributes In Progress Status

| Value | Description |
|---|---|
| 0 | No update pending |
| 1 | Overlay attributes will be updated on the next frame |

**Bit 0 – CHSR** Channel Status

| Value | Description |
|---|---|
| 0 | Layer disabled |
| 1 | Layer enabled |

### 38.7.79 High-End Overlay Interrupt Enable Register

**Name:** LCDC_HEOIER
**Offset:** 0x0000034C
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | VOVR | VDONE | VADD | VDSCR | VDMA | | |
| Access | | W | W | W | W | W | | |
| Reset | | – | – | – | – | – | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | UOVR | UDONE | UADD | UDSCR | UDMA | | |
| Access | | W | W | W | W | W | | |
| Reset | | – | – | – | – | – | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | OVR | DONE | ADD | DSCR | DMA | | |
| Access | | W | W | W | W | W | | |
| Reset | | – | – | – | – | – | | |

**Bit 22 – VOVR** Overflow for V Chrominance Interrupt Enable

**Bit 21 – VDONE** End of List for V Chrominance Interrupt Enable

**Bit 20 – VADD** Head Descriptor Loaded for V Chrominance Interrupt Enable

**Bit 19 – VDSCR** Descriptor Loaded for V Chrominance Interrupt Enable

**Bit 18 – VDMA** End of DMA for V Chrominance Transfer Interrupt Enable

**Bit 14 – UOVR** Overflow for U or UV Chrominance Interrupt Enable

**Bit 13 – UDONE** End of List for U or UV Chrominance Interrupt Enable

**Bit 12 – UADD** Head Descriptor Loaded for U or UV Chrominance Interrupt Enable

**Bit 11 – UDSCR** Descriptor Loaded for U or UV Chrominance Interrupt Enable

**Bit 10 – UDMA** End of DMA Transfer for U or UV Chrominance Interrupt Enable

**Bit 6 – OVR** Overflow Interrupt Enable

**Bit 5 – DONE** End of List Interrupt Enable

**Bit 4 – ADD** Head Descriptor Loaded Interrupt Enable

**Bit 3 – DSCR** Descriptor Loaded Interrupt Enable

**Bit 2 – DMA** End of DMA Transfer Interrupt Enable

### 38.7.80 High-End Overlay Interrupt Disable Register

**Name:** LCDC_HEOIDR
**Offset:** 0x00000350
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | VOVR | VDONE | VADD | VDSCR | VDMA | | |
| Access | | W | W | W | W | W | | |
| Reset | | – | – | – | – | – | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | UOVR | UDONE | UADD | UDSCR | UDMA | | |
| Access | | W | W | W | W | W | | |
| Reset | | – | – | – | – | – | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | OVR | DONE | ADD | DSCR | DMA | | |
| Access | | W | W | W | W | W | | |
| Reset | | – | – | – | – | – | | |

**Bit 22 – VOVR** Overflow for V Chrominance Component Interrupt Disable

**Bit 21 – VDONE** End of List for V Chrominance Component Interrupt Disable

**Bit 20 – VADD** Head Descriptor Loaded for V Chrominance Component Interrupt Disable

**Bit 19 – VDSCR** Descriptor Loaded for V Chrominance Component Interrupt Disable

**Bit 18 – VDMA** End of DMA Transfer for V Chrominance Component Interrupt Disable

**Bit 14 – UOVR** Overflow Interrupt for U or UV Chrominance Component Disable

**Bit 13 – UDONE** End of List Interrupt for U or UV Chrominance Component Disable

**Bit 12 – UADD** Head Descriptor Loaded for U or UV Chrominance Component Interrupt Disable

**Bit 11 – UDSCR** Descriptor Loaded for U or UV Chrominance Component Interrupt Disable

© 2018 Microchip Technology Inc.

**Bit 10 – UDMA** End of DMA Transfer for U or UV Chrominance Component Interrupt Disable

**Bit 6 – OVR** Overflow Interrupt Disable

**Bit 5 – DONE** End of List Interrupt Disable

**Bit 4 – ADD** Head Descriptor Loaded Interrupt Disable

**Bit 3 – DSCR** Descriptor Loaded Interrupt Disable

**Bit 2 – DMA** End of DMA Transfer Interrupt Disable

### 38.7.81 High-End Overlay Interrupt Mask Register

**Name:** LCDC_HEOIMR
**Offset:** 0x00000354
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:
0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | VOVR | VDONE | VADD | VDSCR | VDMA | | |
| Access | | R | R | R | R | R | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | UOVR | UDONE | UADD | UDSCR | UDMA | | |
| Access | | R | R | R | R | R | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | OVR | DONE | ADD | DSCR | DMA | | |
| Access | | R | R | R | R | R | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | | |

**Bit 22 – VOVR** Overflow for V Chrominance Interrupt Mask

**Bit 21 – VDONE** End of List for V Chrominance Component Mask

**Bit 20 – VADD** Head Descriptor Loaded for V Chrominance Component Mask

**Bit 19 – VDSCR** Descriptor Loaded for V Chrominance Component Interrupt Mask

**Bit 18 – VDMA** End of DMA Transfer for V Chrominance Component Interrupt Mask

**Bit 14 – UOVR** Overflow for U Chrominance Interrupt Mask

**Bit 13 – UDONE** End of List for U or UV Chrominance Component Mask

**Bit 12 – UADD** Head Descriptor Loaded for U or UV Chrominance Component Mask

**Bit 11 – UDSCR** Descriptor Loaded for U or UV Chrominance Component Interrupt Mask

**Bit 10 – UDMA** End of DMA Transfer for U or UV Chrominance Component Interrupt Mask

**Bit 6 – OVR** Overflow Interrupt Mask

**Bit 5 – DONE** End of List Interrupt Mask

**Bit 4 – ADD** Head Descriptor Loaded Interrupt Mask

**Bit 3 – DSCR** Descriptor Loaded Interrupt Mask

**Bit 2 – DMA** End of DMA Transfer Interrupt Mask

### 38.7.82 High-End Overlay Interrupt Status Register

**Name:** LCDC_HEOISR
**Offset:** 0x00000358
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | VOVR | VDONE | VADD | VDSCR | VDMA | | |
| Access | | R | R | R | R | R | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | UOVR | UDONE | UADD | UDSCR | UDMA | | |
| Access | | R | R | R | R | R | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | OVR | DONE | ADD | DSCR | DMA | | |
| Access | | R | R | R | R | R | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | | |

**Bit 22 – VOVR** Overflow Detected for V Component

| Value | Description |
|---|---|
| 0 | No overflow occurred since last read of LCDC_HEOISR |
| 1 | An overflow occurred. This flag is reset after a read operation. |

**Bit 21 – VDONE** End of List Detected for V Component

| Value | Description |
|---|---|
| 0 | No End of List condition occurred since last read of LCDC_HEOISR |
| 1 | End of List condition has occurred. This flag is reset after a read operation. |

**Bit 20 – VADD** Head Descriptor Loaded for V Component

| Value | Description |
|---|---|
| 0 | No descriptor has been loaded since last read of LCDC_HEOISR |
| 1 | The descriptor pointed to by the LCDC_HEOVHEAD register has been loaded successfully. This flag is reset after a read operation. |

**Bit 19 – VDSCR** DMA Descriptor Loaded for V Component

| Value | Description |
|---|---|
| 0 | No descriptor has been loaded since last read of LCDC_HEOISR |

| Value | Description |
|---|---|
| 1 | A descriptor has been loaded successfully. This flag is reset after a read operation. |

**Bit 18 – VDMA** End of DMA Transfer for V Component

| Value | Description |
|---|---|
| 0 | No End of Transfer has been detected since last read of LCDC_HEOISR |
| 1 | End of Transfer has been detected. This flag is reset after a read operation. |

**Bit 14 – UOVR** Overflow Detected for U Component

| Value | Description |
|---|---|
| 0 | No overflow occurred since last read of LCDC_HEOISR |
| 1 | An overflow occurred. This flag is reset after a read operation. |

**Bit 13 – UDONE** End of List Detected for U Component

| Value | Description |
|---|---|
| 0 | No End of List condition occurred since last read of LCDC_HEOISR |
| 1 | End of List condition has occurred. This flag is reset after a read operation. |

**Bit 12 – UADD** Head Descriptor Loaded for U Component

| Value | Description |
|---|---|
| 0 | No descriptor has been loaded since last read of LCDC_HEOISR |
| 1 | The descriptor pointed to by the LCDC_HEOUHEAD register has been loaded successfully. This flag is reset after a read operation. |

**Bit 11 – UDSCR** DMA Descriptor Loaded for U Component

| Value | Description |
|---|---|
| 0 | No descriptor has been loaded since last read of LCDC_HEOISR |
| 1 | A descriptor has been loaded successfully. This flag is reset after a read operation. |

**Bit 10 – UDMA** End of DMA Transfer for U Component

| Value | Description |
|---|---|
| 0 | No End of Transfer has been detected since last read of LCDC_HEOISR |
| 1 | End of Transfer has been detected. This flag is reset after a read operation. |

**Bit 6 – OVR** Overflow Detected

| Value | Description |
|---|---|
| 0 | No overflow occurred since last read of LCDC_HEOISR |
| 1 | An overflow occurred. This flag is reset after a read operation. |

**Bit 5 – DONE** End of List Detected

| Value | Description |
|---|---|
| 0 | No End of List condition occurred since last read of LCDC_HEOISR |
| 1 | End of List condition has occurred. This flag is reset after a read operation. |

**Bit 4 – ADD** Head Descriptor Loaded

| Value | Description |
|---|---|
| 0 | No descriptor has been loaded since last read of LCDC_HEOISR |
| 1 | The descriptor pointed to by the LCDC_HEOHEAD register has been loaded successfully. This flag is reset after a read operation. |

**Bit 3 – DSCR** DMA Descriptor Loaded

| Value | Description |
|---|---|
| 0 | No descriptor has been loaded since last read of LCDC_HEOISR |
| 1 | A descriptor has been loaded successfully. This flag is reset after a read operation. |

**Bit 2 – DMA** End of DMA Transfer

| Value | Description |
|---|---|
| 0 | No end of transfer has been detected since last read of LCDC_HEOISR |
| 1 | End of Transfer has been detected. This flag is reset after a read operation. |

### 38.7.83 High-End Overlay DMA Head Register

**Name:** LCDC_HEOHEAD
**Offset:** 0x0000035C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – HEAD[29:0]** DMA Head Pointer
The Head Pointer points to a new descriptor.

### 38.7.84 High-End Overlay DMA Address Register

**Name:** LCDC_HEOADDR
**Offset:** 0x00000360
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** DMA Transfer Start Address
Frame Buffer Base Address.

### 38.7.85 High-End Overlay DMA Control Register

**Name:** LCDC_HEOCTRL
**Offset:** 0x00000364
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DONEIEN | ADDIEN | DSCRIEN | DMAIEN | LFETCH | DFETCH |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 5 – DONEIEN** End of List Interrupt Enable

| Value | Description |
|---|---|
| 0 | End of list interrupt is disabled. |
| 1 | End of list interrupt is enabled. |

**Bit 4 – ADDIEN** Add Head Descriptor to Queue Interrupt Enable

| Value | Description |
|---|---|
| 0 | Transfer descriptor added to queue interrupt is enabled. |
| 1 | Transfer descriptor added to queue interrupt is enabled. |

**Bit 3 – DSCRIEN** Descriptor Loaded Interrupt Enable

| Value | Description |
|---|---|
| 0 | Transfer descriptor loaded interrupt is enabled. |
| 1 | Transfer descriptor loaded interrupt is disabled. |

**Bit 2 – DMAIEN** End of DMA Transfer Interrupt Enable

| Value | Description |
|---|---|
| 0 | DMA transfer completed interrupt is enabled. |
| 1 | DMA transfer completed interrupt is disabled. |

**Bit 1 – LFETCH** Lookup Table Fetch Enable

| Value | Description |
|---|---|
| 0 | Lookup Table DMA fetch is disabled. |
| 1 | Lookup Table DMA fetch is enabled. |

**Bit 0 – DFETCH** Transfer Descriptor Fetch Enable

| Value | Description |
|---|---|
| 0 | Transfer Descriptor fetch is disabled. |
| 1 | Transfer Descriptor fetch is enabled. |

### 38.7.86 High-End Overlay DMA Next Register

**Name:** LCDC_HEONEXT
**Offset:** 0x00000368
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NEXT[31:0]** DMA Descriptor Next Address
The transfer descriptor address must be aligned on a 64-bit boundary.

### 38.7.87 High-End Overlay U-UV DMA Head Register

**Name:** LCDC_HEOUHEAD
**Offset:** 0x0000036C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | UHEAD[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UHEAD[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UHEAD[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UHEAD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – UHEAD[31:0]** DMA Head Pointer
The Head Pointer points to a new descriptor.

### 38.7.88 High-End Overlay U-UV DMA Address Register

**Name:** LCDC_HEOUADDR
**Offset:** 0x00000370
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | UADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – UADDR[31:0]** DMA Transfer Start Address for U or UV Chrominance
U or UV frame buffer address.

### 38.7.89 High-End Overlay U-UV DMA Control Register

**Name:** LCDC_HEOUCTRL
**Offset:** 0x00000374
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | UDONEIEN | UADDIEN | UDSCRIEN | UDMAIEN | | UDFETCH |
| Access | | | R/W | R/W | R/W | R/W | | R/W |
| Reset | | | 0 | 0 | 0 | 0 | | 0 |

**Bit 5 – UDONEIEN** End of List Interrupt Enable

| Value | Description |
|---|---|
| 0 | End of list interrupt is disabled. |
| 1 | End of list interrupt is enabled. |

**Bit 4 – UADDIEN** Add Head Descriptor to Queue Interrupt Enable

| Value | Description |
|---|---|
| 0 | Transfer descriptor added to queue interrupt is enabled. |
| 1 | Transfer descriptor added to queue interrupt is enabled. |

**Bit 3 – UDSCRIEN** Descriptor Loaded Interrupt Enable

| Value | Description |
|---|---|
| 0 | Transfer descriptor loaded interrupt is enabled. |
| 1 | Transfer descriptor loaded interrupt is disabled. |

**Bit 2 – UDMAIEN** End of DMA Transfer Interrupt Enable

| Value | Description |
|---|---|
| 0 | DMA transfer completed interrupt is enabled. |
| 1 | DMA transfer completed interrupt is disabled. |

**Bit 0 – UDFETCH** Transfer Descriptor Fetch Enable

| Value | Description |
|---|---|
| 0 | Transfer Descriptor fetch is disabled. |
| 1 | Transfer Descriptor fetch is enabled. |

### 38.7.90 High-End Overlay U-UV DMA Next Register

**Name:** LCDC_HEOUNEXT
**Offset:** 0x00000378
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | UNEXT[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UNEXT[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UNEXT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNEXT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – UNEXT[31:0]** DMA Descriptor Next Address
The transfer descriptor address must be aligned on a 64-bit boundary.

### 38.7.91 High-End Overlay V DMA Head Register

**Name:** LCDC_HEOVHEAD
**Offset:** 0x0000037C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | VHEAD[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | VHEAD[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | VHEAD[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | VHEAD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – VHEAD[31:0]** DMA Head Pointer
The Head Pointer points to a new descriptor.

### 38.7.92 High-End Overlay V DMA Address Register

**Name:** LCDC_HEOVADDR
**Offset:** 0x00000380
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | VADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | VADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | VADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | VADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – VADDR[31:0]** DMA Transfer Start Address for V Chrominance Frame Buffer Base Address.

### 38.7.93 High-End Overlay V DMA Control Register

**Name:** LCDC_HEOVCTRL
**Offset:** 0x00000384
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | VDONEIEN | VADDIEN | VDSCRIEN | VDMAIEN | | VDFETCH |
| Access | | | R/W | R/W | R/W | R/W | | R/W |
| Reset | | | 0 | 0 | 0 | 0 | | 0 |

**Bit 5 – VDONEIEN** End of List Interrupt Enable

| Value | Description |
|---|---|
| 0 | End of list interrupt is disabled. |
| 1 | End of list interrupt is enabled. |

**Bit 4 – VADDIEN** Add Head Descriptor to Queue Interrupt Enable

| Value | Description |
|---|---|
| 0 | Transfer descriptor added to queue interrupt is enabled. |
| 1 | Transfer descriptor added to queue interrupt is enabled. |

**Bit 3 – VDSCRIEN** Descriptor Loaded Interrupt Enable

| Value | Description |
|---|---|
| 0 | Transfer descriptor loaded interrupt is enabled. |
| 1 | Transfer descriptor loaded interrupt is disabled. |

**Bit 2 – VDMAIEN** End of DMA Transfer Interrupt Enable

| Value | Description |
|---|---|
| 0 | DMA transfer completed interrupt is enabled. |
| 1 | DMA transfer completed interrupt is disabled. |

**Bit 0 – VDFETCH** Transfer Descriptor Fetch Enable

| Value | Description |
|---|---|
| 0 | Transfer Descriptor fetch is disabled. |
| 1 | Transfer Descriptor fetch is enabled. |

### 38.7.94 High-End Overlay V DMA Next Register

**Name:** LCDC_HEOVNEXT
**Offset:** 0x00000388
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | VNEXT[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | VNEXT[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | VNEXT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | VNEXT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – VNEXT[31:0]** DMA Descriptor Next Address
The transfer descriptor address must be aligned on a 64-bit boundary.

### 38.7.95 High-End Overlay Configuration Register 0

**Name:** LCDC_HEOCFG0
**Offset:** 0x0000038C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | LOCKDIS | ROTDIS | | | | DLBO |
| Access | | | R/W | R/W | | | | R/W |
| Reset | | | 0 | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BLENUV[1:0] | | BLEN[1:0] | | | | | SIF |
| Access | R/W | R/W | R/W | R/W | | | | R/W |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

**Bit 13 – LOCKDIS** Hardware Rotation Lock Disable

| Value | Description |
|---|---|
| 0 | AHB lock signal is asserted when a rotation is performed. |
| 1 | AHB lock signal is cleared when a rotation is performed. |

**Bit 12 – ROTDIS** Hardware Rotation Optimization Disable

| Value | Description |
|---|---|
| 0 | Rotation optimization is enabled. |
| 1 | Rotation optimization is disabled. |

**Bit 8 – DLBO** Defined Length Burst Only For Channel Bus Transaction

| Value | Description |
|---|---|
| 0 | Undefined length INCR burst is used for a burst of 2 and 3 beats. |
| 1 | Only Defined Length burst is used (SINGLE, INCR4, INCR8 and INCR16). |

**Bits 7:6 – BLENUV[1:0]** AHB Burst Length for U-V Channel

| Value | Name | Description |
|---|---|---|
| 0 | AHB_SINGLE | AHB Access is started as soon as there is enough space in the FIFO to store one data. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats. |
| 1 | AHB_INCR4 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 4 data. An AHB INCR4 Burst is used. SINGLE, INCR and INCR4 bursts are used. INCR is used for a burst of 2 and 3 beats. |
| 2 | AHB_INCR8 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 8 data. An AHB INCR8 Burst is used. SINGLE, INCR, INCR4 and INCR8 bursts are used. INCR is used for a burst of 2 and 3 beats. |
| 3 | AHB_INCR16 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 16 data. An AHB INCR16 Burst is used. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats. |

**Bits 5:4 – BLEN[1:0]** AHB Burst Length

| Value | Name | Description |
|---|---|---|
| 0 | AHB_BLEN_SINGLE | AHB Access is started as soon as there is enough space in the FIFO to store one data. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats. |
| 1 | AHB_BLEN_INCR4 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 4 data. An AHB INCR4 Burst is used. SINGLE, INCR and INCR4 bursts are used. INCR is used for a burst of 2 and 3 beats. |
| 2 | AHB_BLEN_INCR8 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 8 data. An AHB INCR8 Burst is used. SINGLE, INCR, INCR4 and INCR8 bursts are used. INCR is used for a burst of 2 and 3 beats. |
| 3 | AHB_BLEN_INCR16 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 16 data. An AHB INCR16 Burst is used. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats. |

**Bit 0 – SIF** Source Interface

| Value | Description |
|---|---|
| 0 | Base Layer data is retrieved through AHB interface 0. |
| 1 | Base Layer data is retrieved through AHB interface 1. |

### 38.7.96 High-End Overlay Configuration Register 1

**Name:** LCDC_HEOCFG1
**Offset:** 0x0000038C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | DSCALEOPT | | | YUV422SWP | YUV422ROT |
| Access | | | | R/W | | | R/W | R/W |
| Reset | | | | 0 | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | YUVMODE[3:0] | | | | | | CLUTMODE[1:0] | |
| Access | R/W | R/W | R/W | R/W | | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RGBMODE[3:0] | | | | | | YUVEN | CLUTEN |
| Access | R/W | R/W | R/W | R/W | | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

**Bit 20 – DSCALEOPT** Down Scaling Bandwidth Optimization

| Value | Description |
|---|---|
| 0 | Scaler Optimization is disabled. |
| 1 | Scaler Optimization is enabled; only relevant pixels are retrieved from memory to fill the scaler filter. |

**Bit 17 – YUV422SWP** YUV 4:2:2 Swap

| Value | Description |
|---|---|
| 0 | The two Y components of the YUV 4:2:2 packed data stream are not swapped |
| 1 | The two Y components of the YUV 4:2:2 packed data stream are swapped |

**Bit 16 – YUV422ROT** YUV 4:2:2 Rotation
This bit is relevant only when a rotation angle of 90 degrees or 270 degrees is used.

| Value | Description |
|---|---|
| 0 | Chroma Upsampling kernel is configured to use 0 and 180 degrees algorithm |
| 1 | Chroma Upsampling kernel is configured to use the 4:2:2 Rotation Algorithm. |

**Bits 15:12 – YUVMODE[3:0]** YUV Mode Input Selection

| Value | Name | Description |
|---|---|---|
| 0 | 32BPP_AYCBCR | 32 bpp AYCbCr 444 |
| 1 | 16BPP_YCBCR_MODE0 | 16 bpp Cr(n)Y(n+1)Cb(n)Y(n) 422 |
| 2 | 16BPP_YCBCR_MODE1 | 16 bpp Y(n+1)Cr(n)Y(n)Cb(n) 422 |
| 3 | 16BPP_YCBCR_MODE2 | 16 bpp Cb(n)Y(+1)Cr(n)Y(n) 422 |
| 4 | 16BPP_YCBCR_MODE3 | 16 bpp Y(n+1)Cb(n)Y(n)Cr(n) 422 |
| 5 | 16BPP_YCBCR_SEMIPLANAR | 16 bpp Semiplanar 422 YCbCr |
| 6 | 16BPP_YCBCR_PLANAR | 16 bpp Planar 422 YCbCr |
| 7 | 12BPP_YCBCR_SEMIPLANAR | 12 bpp Semiplanar 420 YCbCr |
| 8 | 12BPP_YCBCR_PLANAR | 12 bpp Planar 420 YCbCr |

**Bits 9:8 – CLUTMODE[1:0]** Color Lookup Table Mode Input Selection

| Value | Name | Description |
|---|---|---|
| 0 | CLUT_1BPP | Color Lookup Table mode set to 1 bit per pixel |
| 1 | CLUT_2BPP | Color Lookup Table mode set to 2 bits per pixel |
| 2 | CLUT_4BPP | Color Lookup Table mode set to 4 bits per pixel |
| 3 | CLUT_8BPP | Color Lookup Table mode set to 8 bits per pixel |

**Bits 7:4 – RGBMODE[3:0]** RGB Mode Input Selection

| Value | Name | Description |
|---|---|---|
| 0 | 12BPP_RGB_444 | 12 bpp RGB 444 |
| 1 | 16BPP_ARGB_4444 | 16 bpp ARGB 4444 |
| 2 | 16BPP_RGBA_4444 | 16 bpp RGBA 4444 |
| 3 | 16BPP_RGB_565 | 16 bpp RGB 565 |
| 4 | 16BPP_TRGB_1555 | 16 bpp TRGB 1555 |
| 5 | 18BPP_RGB_666 | 18 bpp RGB 666 |
| 6 | 18BPP_RGB_666PACKED | 18 bpp RGB 666 PACKED |
| 7 | 19BPP_TRGB_1666 | 19 bpp TRGB 1666 |
| 8 | 19BPP_TRGB_PACKED | 19 bpp TRGB 1666 PACKED |
| 9 | 24BPP_RGB_888 | 24 bpp RGB 888 |
| 10 | 24BPP_RGB_888_PACKED | 24 bpp RGB 888 PACKED |
| 11 | 25BPP_TRGB_1888 | 25 bpp TRGB 1888 |
| 12 | 32BPP_ARGB_8888 | 32 bpp ARGB 8888 |
| 13 | 32BPP_RGBA_8888 | 32 bpp RGBA 8888 |

**Bit 1 – YUVEN** YUV Color Space Enable
Color Space is YUV

| Value | Description |
|---|---|
| 0 | Color space is RGB |
| 1 | Color space is YUV |

**Bit 0 – CLUTEN** Color Lookup Table Mode Enable

| Value | Description |
|---|---|
| 0 | RGB mode is selected. |
| 1 | Color Lookup Table mode is selected. |

### 38.7.97 High-End Overlay Configuration Register 2

**Name:** LCDC_HEOCFG2
**Offset:** 0x00000394
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | YPOS[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YPOS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | XPOS[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPOS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 26:16 – YPOS[10:0]** Vertical Window Position
High-End Overlay Vertical window position.

**Bits 10:0 – XPOS[10:0]** Horizontal Window Position
High-End Overlay Horizontal window position.

### 38.7.98 High-End Overlay Configuration Register 3

**Name:** LCDC_HEOCFG3
**Offset:** 0x00000398
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | YSIZE[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | XSIZE[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 26:16 – YSIZE[10:0]** Vertical Window Size
High-End Overlay window height in pixels. The window height is set to (YSIZE + 1).

The following constraint must be met: YPOS + YSIZE ≤ RPF

**Bits 10:0 – XSIZE[10:0]** Horizontal Window Size
High-End Overlay window width in pixels. The window width is set to (XSIZE + 1).

The following constraint must be met: XPOS + XSIZE ≤ PPL

### 38.7.99 High-End Overlay Configuration Register 4

**Name:** LCDC_HEOCFG4
**Offset:** 0x0000039C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | YMEMSIZE[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YMEMSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | XMEMSIZE[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XMEMSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 26:16 – YMEMSIZE[10:0]** Vertical image Size in Memory
High-End Overlay image height in pixels. The image height is set to (YMEMSIZE + 1).

**Bits 10:0 – XMEMSIZE[10:0]** Horizontal image Size in Memory
High-End Overlay image width in pixels. The image width is set to (XMEMSIZE + 1).

### 38.7.100 High-End Overlay Configuration Register 5

**Name:** LCDC_HEOCFG5
**Offset:** 0x000003A0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – XSTRIDE[31:0]** Horizontal Stride
XSTRIDE represents the memory offset, in bytes, between two rows of the image memory.

### 38.7.101 High-End Overlay Configuration Register 6

**Name:** LCDC_HEOCFG6
**Offset:** 0x000003A4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PSTRIDE[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PSTRIDE[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PSTRIDE[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PSTRIDE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – PSTRIDE[31:0]** Pixel Stride
PSTRIDE represents the memory offset, in bytes, between two pixels of the image memory.

### 38.7.102 High-End Overlay Configuration Register 7

**Name:** LCDC_HEOCFG7
**Offset:** 0x000003A8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | UVXSTRIDE[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UVXSTRIDE[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UVXSTRIDE[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UVXSTRIDE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – UVXSTRIDE[31:0]** UV Horizontal Stride
UVXSTRIDE represents the memory offset, in bytes, between two rows of the image memory.

### 38.7.103 High-End Overlay Configuration Register 8

**Name:** LCDC_HEOCFG8
**Offset:** 0x000003AC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | UVPSTRIDE[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UVPSTRIDE[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UVPSTRIDE[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UVPSTRIDE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – UVPSTRIDE[31:0]** UV Pixel Stride
UVPSTRIDE represents the memory offset, in bytes, between two pixels of the image memory.

### 38.7.104 High-End Overlay Configuration Register 9

**Name:** LCDC_HEOCFG9
**Offset:** 0x000003B0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RDEF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GDEF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BDEF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – RDEF[7:0]** Red Default
Default Red color when the High-End Overlay DMA channel is disabled.

**Bits 15:8 – GDEF[7:0]** Green Default
Default Green color when the High-End Overlay DMA channel is disabled.

**Bits 7:0 – BDEF[7:0]** Blue Default
Default Blue color when the High-End Overlay DMA channel is disabled.

### 38.7.105 High-End Overlay Configuration Register 10

**Name:** LCDC_HEOCFG10
**Offset:** 0x000003B4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RKEY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GKEY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BKEY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – RKEY[7:0]** Red Color Component Chroma Key
Reference Red chroma key used to match the Red color of the current overlay.

**Bits 15:8 – GKEY[7:0]** Green Color Component Chroma Key
Reference Green chroma key used to match the Green color of the current overlay.

**Bits 7:0 – BKEY[7:0]** Blue Color Component Chroma Key
Reference Blue chroma key used to match the Blue color of the current overlay.

### 38.7.106 High-End Overlay Configuration Register 11

**Name:** LCDC_HEOCFG11
**Offset:** 0x000003B8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RMASK[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GMASK[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BMASK[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – RMASK[7:0]** Red Color Component Chroma Key Mask
Red Mask used when the compare function is used. If a bit is set then this bit is compared.

**Bits 15:8 – GMASK[7:0]** Green Color Component Chroma Key Mask
Green Mask used when the compare function is used. If a bit is set then this bit is compared.

**Bits 7:0 – BMASK[7:0]** Blue Color Component Chroma Key Mask
Blue Mask used when the compare function is used. If a bit is set then this bit is compared.

### 38.7.107 High-End Overlay Configuration Register 12

**Name:** LCDC_HEOCFG12
**Offset:** 0x000003BC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | VIDPRI | | DSTKEY | REP | DMA |
| Access | | | | R/W | | R/W | R/W | R/W |
| Reset | | | | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVR | LAEN | GAEN | REVALPHA | ITER | ITER2BL | INV | CRKEY |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – GA[7:0]** Blender Global Alpha
Global alpha blender for the current layer.

**Bit 12 – VIDPRI** Video Priority

| Value | Description |
|---|---|
| 0 | OVR1 layer is above HEO layer. |
| 1 | OVR1 layer is below HEO layer. |

**Bit 10 – DSTKEY** Destination Chroma Keying

| Value | Description |
|---|---|
| 0 | Source Chroma keying is enabled. |
| 1 | Destination Chroma keying is used. |

**Bit 9 – REP** Use Replication logic to expand RGB color to 24 bits

| Value | Description |
|---|---|
| 0 | When the selected pixel depth is less than 24 bpp the pixel is shifted and least significant bits are set to 0. |
| 1 | When the selected pixel depth is less than 24 bpp the pixel is shifted and the least significant bit replicates the msb. |

**Bit 8 – DMA** Blender DMA Layer Enable

| Value | Description |
|---|---|
| 0 | The default color is used on the Overlay 1 Layer. |
| 1 | The DMA channel retrieves the pixels stream from the memory. |

**Bit 7 – OVR** Blender Overlay Layer Enable

| Value | Description |
|---|---|
| 0 | Overlay pixel color is set to the default overlay pixel color. |
| 1 | Overlay pixel color is set to the DMA channel pixel color. |

**Bit 6 – LAEN** Blender Local Alpha Enable

| Value | Description |
|---|---|
| 0 | Local alpha blending coefficient is disabled. |
| 1 | Local alpha blending coefficient is enabled. |

**Bit 5 – GAEN** Blender Global Alpha Enable

| Value | Description |
|---|---|
| 0 | Global alpha blending coefficient is disabled. |
| 1 | Global alpha blending coefficient is enabled. |

**Bit 4 – REVALPHA** Blender Reverse Alpha

| Value | Description |
|---|---|
| 0 | Pixel difference is multiplied by alpha. |
| 1 | Pixel difference is multiplied by 1 - alpha. |

**Bit 3 – ITER** Blender Use Iterated Color

| Value | Description |
|---|---|
| 0 | Pixel difference is set to 0. |
| 1 | Pixel difference is set to the iterated pixel value. |

**Bit 2 – ITER2BL** Blender Iterated Color Enable

| Value | Description |
|---|---|
| 0 | Final adder stage operand is set to 0. |
| 1 | Final adder stage operand is set to the iterated pixel value. |

**Bit 1 – INV** Blender Inverted Blender Output Enable

| Value | Description |
|---|---|
| 0 | Iterated pixel is the blended pixel. |
| 1 | Iterated pixel is the inverted pixel. |

**Bit 0 – CRKEY** Blender Chroma Key Enable

| Value | Description |
|---|---|
| 0 | Chroma key matching is disabled. |
| 1 | Chroma key matching is enabled. |

### 38.7.108 High-End Overlay Configuration Register 13

**Name:** LCDC_HEOCFG13
**Offset:** 0x000003C0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SCALEN | | YFACTOR[13:8] | | | | | |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YFACTOR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | XFACTOR[13:8] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XFACTOR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – SCALEN** Hardware Scaler Enable

| Value | Description |
|---|---|
| 0 | Scaler is disabled |
| 1 | Scaler is enabled. |

**Bits 29:16 – YFACTOR[13:0]** Vertical Scaling Factor
Scaler Vertical Factor.

**Bits 13:0 – XFACTOR[13:0]** Horizontal Scaling Factor
Scaler Horizontal Factor.

### 38.7.109 High-End Overlay Configuration Register 14

**Name:** LCDC_HEOCFG14
**Offset:** 0x000003C4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | CSCYOFF | CSCRV[9:4] | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CSCRV[3:0] | | | | CSCRU[9:6] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CSCRU[5:0] | | | | | | CSCRY[9:8] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CSCRY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 30 – CSCYOFF** Color Space Conversion Offset

| Value | Description |
|---|---|
| 0 | Offset is set to 0. |
| 1 | Offset is set to 16. |

**Bits 29:20 – CSCRV[9:0]** Color Space Conversion V coefficient for Red Component 1:2:7 format
Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

**Bits 19:10 – CSCRU[9:0]** Color Space Conversion U coefficient for Red Component 1:2:7 format
Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

**Bits 9:0 – CSCRY[9:0]** Color Space Conversion Y coefficient for Red Component 1:2:7 format
Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

### 38.7.110 High-End Overlay Configuration Register 15

**Name:** LCDC_HEOCFG15
**Offset:** 0x000003C8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | CSCUOFF | CSCGV[9:4] | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CSCGV[3:0] | | | | CSCGU[9:6] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CSCGU[5:0] | | | | | | CSCGY[9:8] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CSCGY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 30 – CSCUOFF** Color Space Conversion Offset

| Value | Description |
|---|---|
| 0 | Offset is set to 0. |
| 1 | Offset is set to 128. |

**Bits 29:20 – CSCGV[9:0]** Color Space Conversion V coefficient for Green Component 1:2:7 format
Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

**Bits 19:10 – CSCGU[9:0]** Color Space Conversion U coefficient for Green Component 1:2:7 format
Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

**Bits 9:0 – CSCGY[9:0]** Color Space Conversion Y coefficient for Green Component 1:2:7 format
Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

### 38.7.111 High-End Overlay Configuration Register 16

**Name:** LCDC_HEOCFG16
**Offset:** 0x000003CC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | CSCVOFF | CSCBV[9:4] | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CSCBV[3:0] | | | | CSCBU[9:6] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CSCBU[5:0] | | | | | | CSCBY[9:8] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CSCBY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 30 – CSCVOFF** Color Space Conversion Offset

| Value | Description |
|---|---|
| 0 | Offset is set to 0. |
| 1 | Offset is set to 128. |

**Bits 29:20 – CSCBV[9:0]** Color Space Conversion V coefficient for Blue Component 1:2:7 format
Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

**Bits 19:10 – CSCBU[9:0]** Color Space Conversion U coefficient for Blue Component 1:2:7 format
Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

**Bits 9:0 – CSCBY[9:0]** Color Space Conversion Y coefficient for Blue Component 1:2:7 format
Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

### 38.7.112 High-End Overlay Configuration Register 17

**Name:** LCDC_HEOCFG17
**Offset:** 0x000003D0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | XPHI0COEFF3[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | XPHI0COEFF2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | XPHI0COEFF1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPHI0COEFF0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – XPHI0COEFF3[7:0]** Horizontal Coefficient for phase 0 tap 3
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 23:16 – XPHI0COEFF2[7:0]** Horizontal Coefficient for phase 0 tap 2
Coefficient format is 1 magnitude bit and 7 fractional bits.

**Bits 15:8 – XPHI0COEFF1[7:0]** Horizontal Coefficient for phase 0 tap 1
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 7:0 – XPHI0COEFF0[7:0]** Horizontal Coefficient for phase 0 tap 0
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.113 High-End Overlay Configuration Register 18

**Name:** LCDC_HEOCFG18
**Offset:** 0x000003D4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPHI0COEFF4[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – XPHI0COEFF4[7:0]** Horizontal Coefficient for phase 0 tap 4
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.114 High-End Overlay Configuration Register 19

**Name:** LCDC_HEOCFG19
**Offset:** 0x000003D8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | XPHI1COEFF3[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | XPHI1COEFF2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | XPHI1COEFF1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPHI1COEFF0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – XPHI1COEFF3[7:0]** Horizontal Coefficient for phase 1 tap 3
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 23:16 – XPHI1COEFF2[7:0]** Horizontal Coefficient for phase 1 tap 2
Coefficient format is 1 magnitude bit and 7 fractional bits.

**Bits 15:8 – XPHI1COEFF1[7:0]** Horizontal Coefficient for phase 1 tap 1
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 7:0 – XPHI1COEFF0[7:0]** Horizontal Coefficient for phase 1 tap 0
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.115 High-End Overlay Configuration Register 20

**Name:** LCDC_HEOCFG20
**Offset:** 0x000003DC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPHI1COEFF4[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – XPHI1COEFF4[7:0]** Horizontal Coefficient for phase 1 tap 4
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.116 High-End Overlay Configuration Register 21

**Name:** LCDC_HEOCFG21
**Offset:** 0x000003E0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | XPHI2COEFF3[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | XPHI2COEFF2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | XPHI2COEFF1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPHI2COEFF0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – XPHI2COEFF3[7:0]** Horizontal Coefficient for phase 2 tap 3
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 23:16 – XPHI2COEFF2[7:0]** Horizontal Coefficient for phase 2 tap 2
Coefficient format is 1 magnitude bit and 7 fractional bits.

**Bits 15:8 – XPHI2COEFF1[7:0]** Horizontal Coefficient for phase 2 tap 1
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 7:0 – XPHI2COEFF0[7:0]** Horizontal Coefficient for phase 2 tap 0
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.117 High-End Overlay Configuration Register 22

**Name:** LCDC_HEOCFG22
**Offset:** 0x000003E4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPHI2COEFF4[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – XPHI2COEFF4[7:0]** Horizontal Coefficient for phase 2 tap 4
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.118 High-End Overlay Configuration Register 23

**Name:** LCDC_HEOCFG23
**Offset:** 0x000003E8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | XPHI3COEFF3[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | XPHI3COEFF2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | XPHI3COEFF1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPHI3COEFF0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – XPHI3COEFF3[7:0]** Horizontal Coefficient for phase 3 tap 3
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 23:16 – XPHI3COEFF2[7:0]** Horizontal Coefficient for phase 3 tap 2
Coefficient format is 1 magnitude bit and 7 fractional bits.

**Bits 15:8 – XPHI3COEFF1[7:0]** Horizontal Coefficient for phase 3 tap 1
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 7:0 – XPHI3COEFF0[7:0]** Horizontal Coefficient for phase 3 tap 0
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.119 High-End Overlay Configuration Register 24

**Name:** LCDC_HEOCFG24
**Offset:** 0x000003EC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPHI3COEFF4[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – XPHI3COEFF4[7:0]** Horizontal Coefficient for phase 3 tap 4
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.120 High-End Overlay Configuration Register 25

**Name:** LCDC_HEOCFG25
**Offset:** 0x000003F0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | XPHI4COEFF3[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | XPHI4COEFF2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | XPHI4COEFF1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPHI4COEFF0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – XPHI4COEFF3[7:0]** Horizontal Coefficient for phase 4 tap 3
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 23:16 – XPHI4COEFF2[7:0]** Horizontal Coefficient for phase 4 tap 2
Coefficient format is 1 magnitude bit and 7 fractional bits.

**Bits 15:8 – XPHI4COEFF1[7:0]** Horizontal Coefficient for phase 4 tap 1
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 7:0 – XPHI4COEFF0[7:0]** Horizontal Coefficient for phase 4 tap 0
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.121 High-End Overlay Configuration Register 26

**Name:** LCDC_HEOCFG26
**Offset:** 0x000003F4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPHI4COEFF4[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – XPHI4COEFF4[7:0]** Horizontal Coefficient for phase 4 tap 4
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.122 High-End Overlay Configuration Register 27

**Name:** LCDC_HEOCFG27
**Offset:** 0x000003F8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | XPHI5COEFF3[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | XPHI5COEFF2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | XPHI5COEFF1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPHI5COEFF0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – XPHI5COEFF3[7:0]** Horizontal Coefficient for phase 5 tap 3
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 23:16 – XPHI5COEFF2[7:0]** Horizontal Coefficient for phase 5 tap 2
Coefficient format is 1 magnitude bit and 7 fractional bits.

**Bits 15:8 – XPHI5COEFF1[7:0]** Horizontal Coefficient for phase 5 tap 1
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 7:0 – XPHI5COEFF0[7:0]** Horizontal Coefficient for phase 5 tap 0
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.123 High-End Overlay Configuration Register 28

**Name:** LCDC_HEOCFG28
**Offset:** 0x000003FC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPHI5COEFF4[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – XPHI5COEFF4[7:0]** Horizontal Coefficient for phase 5 tap 4
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.124 High-End Overlay Configuration Register 29

**Name:** LCDC_HEOCFG29
**Offset:** 0x00000400
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | XPHI6COEFF3[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | XPHI6COEFF2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | XPHI6COEFF1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPHI6COEFF0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – XPHI6COEFF3[7:0]** Horizontal Coefficient for phase 6 tap 3
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 23:16 – XPHI6COEFF2[7:0]** Horizontal Coefficient for phase 6 tap 2
Coefficient format is 1 magnitude bit and 7 fractional bits.

**Bits 15:8 – XPHI6COEFF1[7:0]** Horizontal Coefficient for phase 6 tap 1
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 7:0 – XPHI6COEFF0[7:0]** Horizontal Coefficient for phase 6 tap 0
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.125 High-End Overlay Configuration Register 30

**Name:** LCDC_HEOCFG30
**Offset:** 0x00000404
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPHI6COEFF4[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – XPHI6COEFF4[7:0]** Horizontal Coefficient for phase 6 tap 4
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.126 High-End Overlay Configuration Register 31

**Name:** LCDC_HEOCFG31
**Offset:** 0x00000408
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | XPHI7COEFF3[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | XPHI7COEFF2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | XPHI7COEFF1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPHI7COEFF0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – XPHI7COEFF3[7:0]** Horizontal Coefficient for phase 7 tap 3
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 23:16 – XPHI7COEFF2[7:0]** Horizontal Coefficient for phase 7 tap 2
Coefficient format is 1 magnitude bit and 7 fractional bits.

**Bits 15:8 – XPHI7COEFF1[7:0]** Horizontal Coefficient for phase 7 tap 1
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 7:0 – XPHI7COEFF0[7:0]** Horizontal Coefficient for phase 7 tap 0
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.127 High-End Overlay Configuration Register 32

**Name:** LCDC_HEOCFG32
**Offset:** 0x0000040C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPHI7COEFF4[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – XPHI7COEFF4[7:0]** Horizontal Coefficient for phase 7 tap 4
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.128 High-End Overlay Configuration Register 33

**Name:** LCDC_HEOCFG33
**Offset:** 0x00000410
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YPHI0COEFF2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | YPHI0COEFF1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | YPHI0COEFF0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – YPHI0COEFF2[7:0]** Vertical Coefficient for phase 0 tap 2
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 15:8 – YPHI0COEFF1[7:0]** Vertical Coefficient for phase 0 tap 1
Coefficient format is 1 magnitude bit and 7 fractional bits.

**Bits 7:0 – YPHI0COEFF0[7:0]** Vertical Coefficient for phase 0 tap 0
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.129 High-End Overlay Configuration Register 34

**Name:** LCDC_HEOCFG34
**Offset:** 0x00000414
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YPHI1COEFF2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | YPHI1COEFF1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | YPHI1COEFF0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – YPHI1COEFF2[7:0]** Vertical Coefficient for phase 1 tap 2
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 15:8 – YPHI1COEFF1[7:0]** Vertical Coefficient for phase 1 tap 1
Coefficient format is 1 magnitude bit and 7 fractional bits.

**Bits 7:0 – YPHI1COEFF0[7:0]** Vertical Coefficient for phase 1 tap 0
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.130 High-End Overlay Configuration Register 35

**Name:** LCDC_HEOCFG35
**Offset:** 0x00000418
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YPHI2COEFF2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | YPHI2COEFF1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | YPHI2COEFF0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – YPHI2COEFF2[7:0]** Vertical Coefficient for phase 2 tap 2
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 15:8 – YPHI2COEFF1[7:0]** Vertical Coefficient for phase 2 tap 1
Coefficient format is 1 magnitude bit and 7 fractional bits.

**Bits 7:0 – YPHI2COEFF0[7:0]** Vertical Coefficient for phase 2 tap 0
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.131 High-End Overlay Configuration Register 36

**Name:** LCDC_HEOCFG36
**Offset:** 0x0000041C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YPHI3COEFF2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | YPHI3COEFF1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | YPHI3COEFF0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – YPHI3COEFF2[7:0]** Vertical Coefficient for phase 3 tap 2
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 15:8 – YPHI3COEFF1[7:0]** Vertical Coefficient for phase 3 tap 1
Coefficient format is 1 magnitude bit and 7 fractional bits.

**Bits 7:0 – YPHI3COEFF0[7:0]** Vertical Coefficient for phase 3 tap 0
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.132 High-End Overlay Configuration Register 37

**Name:** LCDC_HEOCFG37
**Offset:** 0x00000420
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YPHI4COEFF2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | YPHI4COEFF1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | YPHI4COEFF0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – YPHI4COEFF2[7:0]** Vertical Coefficient for phase 4 tap 2
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 15:8 – YPHI4COEFF1[7:0]** Vertical Coefficient for phase 4 tap 1
Coefficient format is 1 magnitude bit and 7 fractional bits.

**Bits 7:0 – YPHI4COEFF0[7:0]** Vertical Coefficient for phase 4 tap 0
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.133 High-End Overlay Configuration Register 38

**Name:** LCDC_HEOCFG38
**Offset:** 0x00000424
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YPHI5COEFF2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | YPHI5COEFF1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | YPHI5COEFF0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – YPHI5COEFF2[7:0]** Vertical Coefficient for phase 5 tap 2
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 15:8 – YPHI5COEFF1[7:0]** Vertical Coefficient for phase 5 tap 1
Coefficient format is 1 magnitude bit and 7 fractional bits.

**Bits 7:0 – YPHI5COEFF0[7:0]** Vertical Coefficient for phase 5 tap 0
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.134 High-End Overlay Configuration Register 39

**Name:** LCDC_HEOCFG39
**Offset:** 0x00000428
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YPHI6COEFF2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | YPHI6COEFF1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | YPHI6COEFF0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – YPHI6COEFF2[7:0]** Vertical Coefficient for phase 6 tap 2
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 15:8 – YPHI6COEFF1[7:0]** Vertical Coefficient for phase 6 tap 1
Coefficient format is 1 magnitude bit and 7 fractional bits.

**Bits 7:0 – YPHI6COEFF0[7:0]** Vertical Coefficient for phase 6 tap 0
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.135 High-End Overlay Configuration Register 40

**Name:** LCDC_HEOCFG40
**Offset:** 0x0000042C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YPHI7COEFF2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | YPHI7COEFF1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | YPHI7COEFF0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – YPHI7COEFF2[7:0]** Vertical Coefficient for phase 7 tap 2
Coefficient format is 1 sign bit and 7 fractional bits.

**Bits 15:8 – YPHI7COEFF1[7:0]** Vertical Coefficient for phase 7 tap 1
Coefficient format is 1 magnitude bit and 7 fractional bits.

**Bits 7:0 – YPHI7COEFF0[7:0]** Vertical Coefficient for phase 7 tap 0
Coefficient format is 1 sign bit and 7 fractional bits.

### 38.7.136 High-End Overlay Configuration Register 41

**Name:** LCDC_HEOCFG41
**Offset:** 0x00000430
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | YPHIDEF[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | XPHIDEF[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 18:16 – YPHIDEF[2:0]** Vertical Filter Phase Offset
XPHIDEF defines the index of the first coefficient set used when the vertical resampling operation is started.

**Bits 2:0 – XPHIDEF[2:0]** Horizontal Filter Phase Offset
XPHIDEF defines the index of the first coefficient set used when the horizontal resampling operation is started.

### 38.7.137 Post Processing Channel Enable Register

**Name:** LCDC_PPCHER
**Offset:** 0x00000540
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | A2QEN | UPDATEEN | CHEN |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

**Bit 2 – A2QEN** Add To Queue Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Indicates that a valid descriptor has been written to memory, its memory location should be written to the DMA head pointer. The A2QSR status bit is set to one, and it is reset by hardware as soon as the descriptor pointed to by the DMA head pointer is added to the list. |

**Bit 1 – UPDATEEN** Update Overlay Attributes Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Updates windows attributes on the next start of frame. |

**Bit 0 – CHEN** Channel Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Enables the DMA channel |

### 38.7.138 Post Processing Channel Disable Register

**Name:** LCDC_PPCHDR
**Offset:** 0x00000544
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 8 – CHRST** Channel Reset

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Resets the layer immediately. The frame is aborted. |

**Bit 0 – CHDIS** Channel Disable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Disables the layer at the end of the current frame. The frame is completed. |

### 38.7.139 Post Processing Channel Status Register

**Name:** LCDC_PPCHSR
**Offset:** 0x00000548
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | A2QSR | UPDATESR | CHSR |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – A2QSR** Add To Queue Status

| Value | Description |
|---|---|
| 0 | Add to queue not pending |
| 1 | Add to queue pending |

**Bit 1 – UPDATESR** Update Overlay Attributes In Progress Status

| Value | Description |
|---|---|
| 0 | No update pending |
| 1 | Overlay attributes will be updated on the next frame |

**Bit 0 – CHSR** Channel Status

| Value | Description |
|---|---|
| 0 | Layer disabled |
| 1 | Layer enabled |

### 38.7.140 Post Processing Interrupt Enable Register

**Name:** LCDC_PPIER
**Offset:** 0x0000054C
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DONE | ADD | DSCR | DMA | | |
| Access | | | W | W | W | W | | |
| Reset | | | – | – | – | – | | |

**Bit 5 – DONE** End of List Interrupt Enable

**Bit 4 – ADD** Head Descriptor Loaded Interrupt Enable

**Bit 3 – DSCR** Descriptor Loaded Interrupt Enable

**Bit 2 – DMA** End of DMA Transfer Interrupt Enable

### 38.7.141 Post Processing Interrupt Disable Register

**Name:** LCDC_PPIDR
**Offset:** 0x00000550
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DONE | ADD | DSCR | DMA | | |
| Access | | | W | W | W | W | | |
| Reset | | | – | – | – | – | | |

**Bit 5 – DONE** End of List Interrupt Disable

**Bit 4 – ADD** Head Descriptor Loaded Interrupt Disable

**Bit 3 – DSCR** Descriptor Loaded Interrupt Disable

**Bit 2 – DMA** End of DMA Transfer Interrupt Disable

### 38.7.142 Post Processing Interrupt Mask Register

**Name:** LCDC_PPIMR
**Offset:** 0x00000554
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:
0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DONE | ADD | DSCR | DMA | | |
| Access | | | R | R | R | R | | |
| Reset | | | 0 | 0 | 0 | 0 | | |

**Bit 5 – DONE** End of List Interrupt Mask

**Bit 4 – ADD** Head Descriptor Loaded Interrupt Mask

**Bit 3 – DSCR** Descriptor Loaded Interrupt Mask

**Bit 2 – DMA** End of DMA Transfer Interrupt Mask

### 38.7.143 Post Processing Interrupt Status Register

**Name:** LCDC_PPISR
**Offset:** 0x00000558
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DONE | ADD | DSCR | DMA | | |
| Access | | | R | R | R | R | | |
| Reset | | | 0 | 0 | 0 | 0 | | |

**Bit 5 – DONE** End of List Detected

| Value | Description |
|---|---|
| 0 | No End of List condition has occurred since last read of LCDC_PPISR |
| 1 | End of List condition has occurred. This flag is reset after a read operation. |

**Bit 4 – ADD** Head Descriptor Loaded

| Value | Description |
|---|---|
| 0 | No descriptor has been loaded since last read of LCDC_PPISR |
| 1 | The descriptor pointed to by the LCDC_PPHEAD register has been loaded successfully. This flag is reset after a read operation. |

**Bit 3 – DSCR** DMA Descriptor Loaded

| Value | Description |
|---|---|
| 0 | No descriptor has been loaded since last read of LCDC_PPISR |
| 1 | A descriptor has been loaded successfully. This flag is reset after a read operation. |

**Bit 2 – DMA** End of DMA Transfer

| Value | Description |
|---|---|
| 0 | No End of Transfer has been detected since last read of LCDC_PPISR |

| Value | Description |
|---|---|
| 1 | End of Transfer has been detected. This flag is reset after a read operation. |

### 38.7.144 Post Processing Head Register

**Name:** LCDC_PPHEAD
**Offset:** 0x0000055C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | HEAD[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – HEAD[29:0]** DMA Head Pointer
The Head Pointer points to a new descriptor.

### 38.7.145 Post Processing Address Register

**Name:** LCDC_PPADDR
**Offset:** 0x00000560
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** DMA Transfer Start Address
Post Processing Destination frame buffer address.

### 38.7.146 Post Processing Control Register

**Name:** LCDC_PPCTRL
**Offset:** 0x00000564
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DONEIEN | ADDIEN | DSCRIEN | DMAIEN | | DFETCH |
| Access | | | R/W | R/W | R/W | R/W | | R/W |
| Reset | | | 0 | 0 | 0 | 0 | | 0 |

**Bit 5 – DONEIEN** End of List Interrupt Enable

| Value | Description |
|---|---|
| 0 | End of list interrupt is disabled. |
| 1 | End of list interrupt is enabled. |

**Bit 4 – ADDIEN** Add Head Descriptor to Queue Interrupt Enable

| Value | Description |
|---|---|
| 0 | Transfer descriptor added to queue interrupt is enabled. |
| 1 | Transfer descriptor added to queue interrupt is enabled. |

**Bit 3 – DSCRIEN** Descriptor Loaded Interrupt Enable

| Value | Description |
|---|---|
| 0 | Transfer descriptor loaded interrupt is enabled. |
| 1 | Transfer descriptor loaded interrupt is disabled. |

**Bit 2 – DMAIEN** End of DMA Transfer Interrupt Enable

| Value | Description |
|---|---|
| 0 | DMA transfer completed interrupt is enabled. |
| 1 | DMA transfer completed interrupt is disabled. |

**Bit 0 – DFETCH** Transfer Descriptor Fetch Enable

| Value | Description |
|---|---|
| 0 | Transfer Descriptor fetch is disabled. |
| 1 | Transfer Descriptor fetch is enabled. |

### 38.7.147 Post Processing Next Register

**Name:** LCDC_PPNEXT
**Offset:** 0x00000568
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NEXT[31:0]** DMA Descriptor Next Address
The transfer descriptor address must be aligned on a 64-bit boundary.

### 38.7.148 Post Processing Configuration Register 0

**Name:** LCDC_PPCFG0
**Offset:** 0x0000056C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DLBO |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | BLEN[1:0] | | | | | SIF |
| Access | | | R/W | R/W | | | | R/W |
| Reset | | | 0 | 0 | | | | 0 |

**Bit 8 – DLBO** Defined Length Burst Only For Channel Bus Transaction

| Value | Description |
|---|---|
| 0 | Undefined length INCR burst is used for 2 and 3 beats burst. |
| 1 | Only Defined Length burst is used (SINGLE, INCR4, INCR8 and INCR16). |

**Bits 5:4 – BLEN[1:0]** AHB Burst Length

| Value | Name | Description |
|---|---|---|
| 0 | AHB_BLEN_SINGLE | AHB Access is started as soon as there is enough space in the FIFO to store one data. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats. |
| 1 | AHB_BLEN_INCR4 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 4 data. An AHB INCR4 Burst is used. SINGLE, INCR and INCR4 bursts are used. INCR is used for a burst of 2 and 3 beats. |
| 2 | AHB_BLEN_INCR8 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 8 data. An AHB INCR8 Burst is used. SINGLE, INCR, INCR4 and INCR8 bursts are used. INCR is used for a burst of 2 and 3 beats. |

| Value | Name | Description |
|---|---|---|
| 3 | AHB_BLEN_INCR16 | AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 16 data. An AHB INCR16 Burst is used. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats. |

**Bit 0 – SIF** Source Interface

| Value | Description |
|---|---|
| 0 | Base Layer data is retrieved through AHB interface 0. |
| 1 | Base Layer data is retrieved through AHB interface 1. |

### 38.7.149 Post Processing Configuration Register 1

**Name:** LCDC_PPCFG1
**Offset:** 0x00000570
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | ITUBT601 | | PPMODE[2:0] | | |
| Access | | | | R/W | | R/W | R/W | R/W |
| Reset | | | | 0 | | 0 | 0 | 0 |

**Bit 4 – ITUBT601** Color Space Conversion Luminance

| Value | Description |
|---|---|
| 0 | Luminance and chrominance range is [0;255] |
| 1 | Luminance values are clamped to [16;235] range. Chrominance values are clamped to [16;240] range. |

**Bits 2:0 – PPMODE[2:0]** Post Processing Output Format Selection

| Value | Name | Description |
|---|---|---|
| 0 | PPMODE_RGB_16BPP | RGB 16 bpp |
| 1 | PPMODE_RGB_24BPP_PACKED | RGB 24 bpp PACKED |
| 2 | PPMODE_RGB_24BPP_UNPACKED | RGB 24 bpp UNPACKED |
| 3 | PPMODE_YCBCR_422_MODE0 | YCbCr 422 16 bpp (Mode 0) |
| 4 | PPMODE_YCBCR_422_MODE1 | YCbCr 422 16 bpp (Mode 1) |
| 5 | PPMODE_YCBCR_422_MODE2 | YCbCr 422 16 bpp (Mode 2) |
| 6 | PPMODE_YCBCR_422_MODE3 | YCbCr 422 16 bpp (Mode 3) |

### 38.7.150 Post Processing Configuration Register 2

**Name:** LCDC_PPCFG2
**Offset:** 0x00000574
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XSTRIDE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – XSTRIDE[31:0]** Horizontal Stride
XSTRIDE represents the memory offset, in bytes, between two rows of the image memory.

### 38.7.151 Post Processing Configuration Register 3

**Name:** LCDC_PPCFG3
**Offset:** 0x00000578
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | CSCYOFF | CSCYB[9:4] | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CSCYB[3:0] | | | | CSCYG[9:6] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CSCYG[5:0] | | | | | | CSCYR[9:8] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CSCYR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 30 – CSCYOFF** Color Space Conversion Luminance Offset

| Value | Description |
|---|---|
| 0 | The Yoff parameter value is set to 0. |
| 1 | The Yoff parameter value is set to 16. |

**Bits 29:20 – CSCYB[9:0]** Color Space Conversion B coefficient for Luminance component, signed format, step set to 1/1024
Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

**Bits 19:10 – CSCYG[9:0]** Color Space Conversion G coefficient for Luminance component, signed format, step set to 1/512
Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

**Bits 9:0 – CSCYR[9:0]** Color Space Conversion R coefficient for Luminance component, signed format, step set to 1/1024
Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

### 38.7.152 Post Processing Configuration Register 4

**Name:** LCDC_PPCFG4
**Offset:** 0x0000057C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | CSCUOFF | CSCUB[9:4] | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CSCUB[3:0] | | | | CSCUG[9:6] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CSCUG[5:0] | | | | | | CSCUR[9:8] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CSCUR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 30 – CSCUOFF** Color Space Conversion Chrominance B Offset

| Value | Description |
|---|---|
| 0 | The Cboff parameter value is set to 0. |
| 1 | The Cboff parameter value is set to 128. |

**Bits 29:20 – CSCUB[9:0]** Color Space Conversion B coefficient for Chrominance B component, signed format. (step 1/512)
Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

**Bits 19:10 – CSCUG[9:0]** Color Space Conversion G coefficient for Chrominance B component, signed format. (step 1/512)
Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

**Bits 9:0 – CSCUR[9:0]** Color Space Conversion R coefficient for Chrominance B component, signed format. (step 1/1024)
Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

### 38.7.153 Post Processing Configuration Register 5

**Name:** LCDC_PPCFG5
**Offset:** 0x00000580
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | CSCVOFF | CSCVB[9:4] | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CSCVB[3:0] | | | | CSCVG[9:6] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CSCVG[5:0] | | | | | | CSCVR[9:8] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CSCVR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 30 – CSCVOFF** Color Space Conversion Chrominance R Offset

| Value | Description |
|---|---|
| 0 | The Croff parameter value is set to 0. |
| 1 | The Croff parameter value is set to 128. |

**Bits 29:20 – CSCVB[9:0]** Color Space Conversion B coefficient for Chrominance R component, signed format. (step 1/1024)
Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

**Bits 19:10 – CSCVG[9:0]** Color Space Conversion G coefficient for Chrominance R component, signed format. (step 1/512)
Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

**Bits 9:0 – CSCVR[9:0]** Color Space Conversion R coefficient for Chrominance R component, signed format. (step 1/1024)
Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

### 38.7.154 Base CLUT Register x

**Name:** LCDC_BASECLUTx
**Offset:** 0x0600 + x*0x04 [x=0..255]
**Reset:** 0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RCLUT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GCLUT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BCLUT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – RCLUT[7:0]** Red Color Entry
This field indicates the 8-bit width Red color of the color lookup table.

**Bits 15:8 – GCLUT[7:0]** Green Color Entry
This field indicates the 8-bit width Green color of the color lookup table.

**Bits 7:0 – BCLUT[7:0]** Blue Color Entry
This field indicates the 8-bit width Blue color of the color lookup table.

### 38.7.155 Overlay 1 CLUT Register x

**Name:** LCDC_OVR1CLUTx
**Offset:** 0x0A00 + x*0x04 [x=0..255]
**Reset:** 0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ACLUT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RCLUT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GCLUT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BCLUT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – ACLUT[7:0]** Alpha Color Entry
This field indicates the 8-bit width Alpha channel of the color lookup table.

**Bits 23:16 – RCLUT[7:0]** Red Color Entry
This field indicates the 8-bit width Red color of the color lookup table.

**Bits 15:8 – GCLUT[7:0]** Green Color Entry
This field indicates the 8-bit width Green color of the color lookup table.

**Bits 7:0 – BCLUT[7:0]** Blue Color Entry
This field indicates the 8-bit width Blue color of the color lookup table.

### 38.7.156 Overlay 2 CLUT Register x

**Name:** LCDC_OVR2CLUTx
**Offset:** 0x0E00 + x*0x04 [x=0..255]
**Reset:** 0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ACLUT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RCLUT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GCLUT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BCLUT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – ACLUT[7:0]** Alpha Color Entry
This field indicates the 8-bit width Alpha channel of the color lookup table.

**Bits 23:16 – RCLUT[7:0]** Red Color Entry
This field indicates the 8-bit width Red color of the color lookup table.

**Bits 15:8 – GCLUT[7:0]** Green Color Entry
This field indicates the 8-bit width Green color of the color lookup table.

**Bits 7:0 – BCLUT[7:0]** Blue Color Entry
This field indicates the 8-bit width Blue color of the color lookup table.

### 38.7.157 High-End Overlay CLUT Register x

**Name:** LCDC_HEOCLUTx
**Offset:** 0x1200 + x*0x04 [x=0..255]
**Reset:** 0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ACLUT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RCLUT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GCLUT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BCLUT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – ACLUT[7:0]** Alpha Color Entry
This field indicates the 8-bit width Alpha channel of the color lookup table.

**Bits 23:16 – RCLUT[7:0]** Red Color Entry
This field indicates the 8-bit width Red color of the color lookup table.

**Bits 15:8 – GCLUT[7:0]** Green Color Entry
This field indicates the 8-bit width Green color of the color lookup table.

**Bits 7:0 – BCLUT[7:0]** Blue Color Entry
This field indicates the 8-bit width Blue color of the color lookup table.

# 39. Ethernet MAC (GMAC)

## 39.1 Description

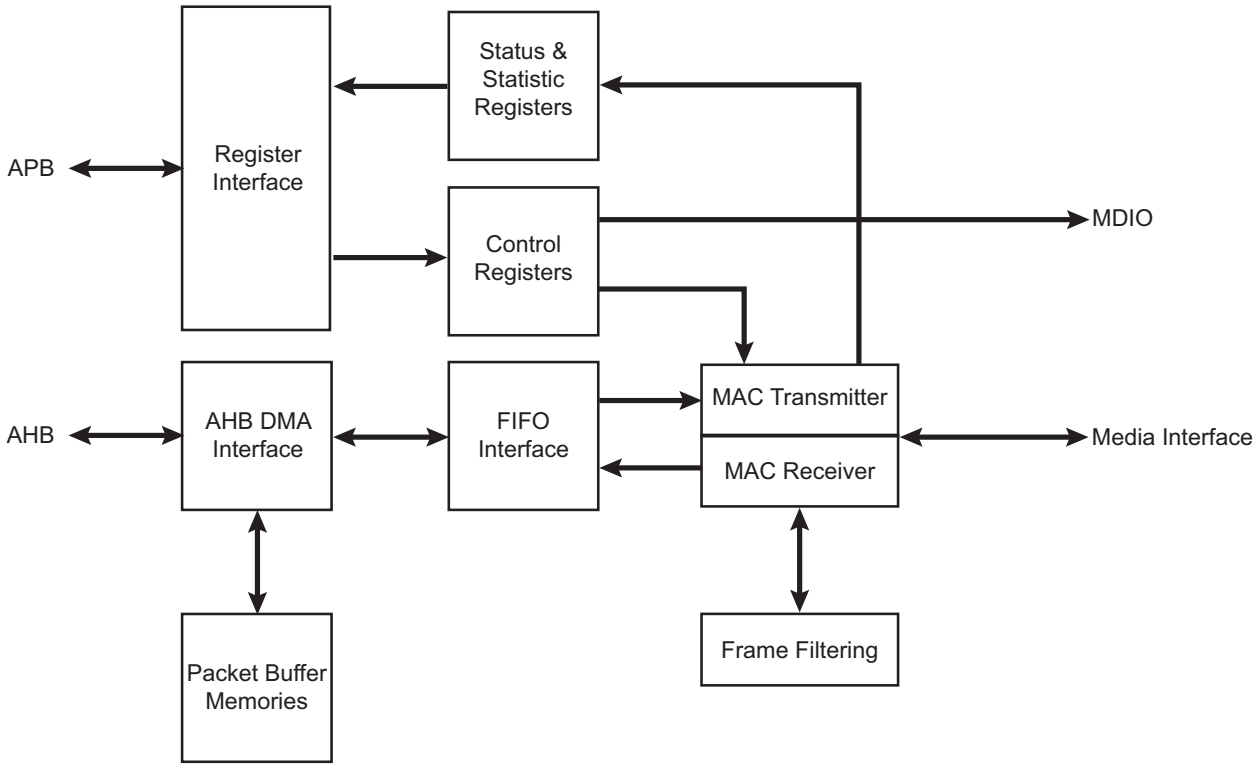
The Ethernet MAC (GEMAC) module implements a 10/100 Mbps Ethernet MAC compatible with the IEEE 802.3 standard. The GEMAC can operate in either half or full duplex mode at all supported speeds. The GMAC Network Configuration Register is used to select the speed, duplex mode and interface type (MII, RMII).

## 39.2 Embedded Characteristics

- Compatible with IEEE Standard 802.3
- 10, 100 Mbps Operation
- Full and Half Duplex Operation at all Supported Speeds of Operation
- Statistics Counter Registers for RMON/MIB
- MII/RMII Interface to the Physical Layer
- Integrated Physical Coding
- Direct Memory Access (DMA) Interface to External Memory
- Support for 3 Priority Queues
- 8 Kbytes Transmit RAM and 4 Kbytes Receive RAM (refer to Table 39-4 for queue-specific sizes)
- Programmable Burst Length and Endianism for DMA
- Interrupt Generation to Signal Receive and Transmit Completion, Errors or Other Events
- Automatic Pad and Cyclic Redundancy Check (CRC) Generation on Transmitted Frames
- Automatic Discard of Frames Received with Errors
- Receive and Transmit IP, TCP and UDP Checksum Offload. Both IPv4 and IPv6 Packet Types Supported
- Address Checking Logic for Four Specific 48-bit Addresses, Four Type IDs, Promiscuous Mode, Hash Matching of Unicast and Multicast Destination Addresses and Wake-on-LAN
- Management Data Input/Output (MDIO) Interface for Physical Layer Management
- Support for Jumbo Frames up to 10240 Bytes
- Full Duplex Flow Control with Recognition of Incoming Pause Frames and Hardware Generation of Transmitted Pause Frames
- Half Duplex Flow Control by Forcing Collisions on Incoming Frames
- Support for 802.1Q VLAN Tagging with Recognition of Incoming VLAN and Priority Tagged Frames
- Support for 802.1Qbb Priority-based Flow Control
- Programmable Inter Packet Gap (IPG) Stretch
- Recognition of IEEE 1588 PTP Frames
- IEEE 1588 Timestamp Unit (TSU)
- Support for 802.1AS Timing and Synchronization
- Supports 802.1Qav Traffic Shaping on Two Highest Priority Queues

## 39.3 Block Diagram

**Figure 39-1. Block Diagram**




## 39.4 Signal Interfaces

The GEMAC includes the following signal interfaces:

- MII, RMII to an external PHY
- MDIO interface for external PHY management
- Slave APB interface for accessing GEMAC registers
- Master AHB interface for memory access
- GTSUCOMP signal for TSU timer count value comparison

**Table 39-1. GEMAC Connections in Different Modes**

| Signal Name | Function | MII | RMII |
|---|---|---|---|
| GTXCK[(1)] | Transmit Clock or Reference Clock | TXCK | REFCK |
| GTXEN | Transmit Enable | TXEN | TXEN |
| GTX[3..0] | Transmit Data | TXD[3:0] | TXD[1:0] |
| GTXER | Transmit Coding Error | TXER | Not Used |
| GRXCK | Receive Clock | RXCK | Not Used |
| GRXDV | Receive Data Valid | RXDV | CRSDV |
| GRX[3..0] | Receive Data | RXD[3:0] | RXD[1:0] |

| .....continued | | | |
|---|---|---|---|
| **Signal Name** | **Function** | **MII** | **RMII** |
| GRXER | Receive Error | RXER | RXER |
| GCRS | Carrier Sense and Data Valid | CRS | Not Used |
| GCOL | Collision Detect | COL | Not Used |
| GMDC | Management Data Clock | MDC | MDC |
| GMDIO | Management Data Input/Output | MDIO | MDIO |

**Note:**

1. Input only. GTXCK must be provided with a 25 MHz / 50 MHz external crystal oscillator for MII / RMII interfaces, respectively.

## 39.5 Product Dependencies

### 39.5.1 I/O Lines

The pins used for interfacing the GEMAC may be multiplexed with PIO lines. The programmer must first program the PIO Controller to assign the pins to their peripheral function. If I/O lines of the GEMAC are not used by the application, they can be used for other purposes by the PIO Controller.

### 39.5.2 Power Management

The GEMAC is not continuously clocked. The user must first enable the GEMAC clock in the Power Management Controller before using it.

### 39.5.3 Interrupt Sources

The GEMAC interrupt line is connected to one of the internal sources of the interrupt controller. Using the GEMAC interrupt requires prior programming of the interrupt controller.

The GEMAC features 3 interrupt sources. Refer to the table “Peripheral Identifiers” in the section “Peripherals” for the interrupt numbers for GEMAC priority queues.

## 39.6 Functional Description

### 39.6.1 Media Access Controller

The Media Access Controller (MAC) transmit block takes data from FIFO, adds preamble and, if necessary, pad and frame check sequence (FCS). Both half duplex and full duplex Ethernet modes of operation are supported. When operating in half duplex mode, the MAC transmit block generates data according to the carrier sense multiple access with collision detect (CSMA/CD) protocol. The start of transmission is deferred if carrier sense (CRS) is active. If collision (COL) becomes active during transmission, a jam sequence is asserted and the transmission is retried after a random backoff. The CRS and COL signals have no effect in full duplex mode.

The MAC receive block checks for valid preamble, FCS, alignment and length, and presents received frames to the MAC address checking block and FIFO. Software can configure the GMAC to receive jumbo frames up to 10240 bytes. It can optionally strip CRC from the received frame prior to transfer to FIFO.

The address checker recognizes four specific 48-bit addresses, can recognize four different type ID values, and contains a 64-bit Hash register for matching multicast and unicast addresses as required. It can recognize the broadcast address of all ones and copy all frames. The MAC can also reject all frames that are not VLAN tagged and recognize Wake on LAN events.

The MAC receive block supports offloading of IP, TCP and UDP checksum calculations (both IPv4 and IPv6 packet types supported), and can automatically discard bad checksum frames.

### 39.6.2 1588 Timestamp Unit

The 1588 timestamp unit (TSU) is implemented as a 94-bit timer.

The 48 upper bits [93:46] of the timer count seconds and are accessible in the GMAC 1588 Timer Seconds High Register (GMAC_TSH) and GMAC 1588 Timer Seconds Low Register (GMAC_TSL). The 30 lower bits [45:16] of the timer count nanoseconds and are accessible in the GMAC 1588 Timer Nanoseconds Register (GMAC_TN). The lowest 16 bits [15:0] of the timer count sub-nanoseconds.

The 46 lower bits roll over when they have counted to one second. The timer increments by a programmable period (to approximately 15.2 femtoseconds resolution) with each MCK period and can also be adjusted in 1ns resolution (incremented or decremented) through APB register accesses.

### 39.6.3 AHB Direct Memory Access Interface

The GEMAC DMA controller is connected to the MAC FIFO interface and provides a scatter-gather type capability for packet data storage.

The DMA implements packet buffering where dual-port memories are used to buffer multiple frames.

#### 39.6.3.1 Packet Buffer DMA

- Easier to guarantee maximum line rate due to the ability to store multiple frames in the packet buffer, where the number of frames is limited by the amount of packet buffer memory and Ethernet frame size
- Full store and forward, or partial store and forward programmable options (partial store will cater for shorter latency requirements)
- Support for Transmit TCP/IP checksum offload
- Support for priority queueing
- When a collision on the line occurs during transmission, the packet will be automatically replayed directly from the packet buffer memory rather than having to re-fetch through the AHB (full store and forward ONLY)
- Received error packets are automatically dropped before any of the packet is presented to the AHB (full store and forward ONLY), thus reducing AHB activity
- Supports manual RX packet flush capabilities
- Optional RX packet flush when there is lack of AHB resource

#### 39.6.3.2 Partial Store and Forward Using Packet Buffer DMA

The DMA uses SRAM-based packet buffers, and can be programmed into a low latency mode, known as Partial Store and Forward. This allows for a reduced latency as the full packet is not buffered before forwarding. Note that this option is only available when the device is configured for full duplex operation.

This feature is enabled via the programmable TX and RX Partial Store and Forward registers. When the transmit Partial Store and Forward mode is activated, the transmitter will only begin to forward the packet to the MAC when there is enough packet data stored in the packet buffer. Likewise, when the receive Partial Store and Forward mode is activated, the receiver will only begin to forward the packet to the AHB when enough packet data is stored in the packet buffer. The amount of packet data required to activate

the forwarding process is programmable via watermark registers which are located at the same address as the partial store and forward enable bits.

Note that the minimum operational value for the TX partial store and forward watermark is 20. There is no operational limit for the RX partial store and forward watermark. Enabling partial store and forward is a useful means to reduce latency, but there are performance implications.

The GEMAC DMA uses separate transmit and receive lists of buffer descriptors, with each descriptor describing a buffer area in memory. This allows Ethernet packets to be broken up and scattered around the AHB memory space.

#### 39.6.3.3 Receive AHB Buffers

Received frames, optionally including FCS, are written to receive AHB buffers stored in memory. The receive buffer depth is programmable in the range of 64 bytes to 16 Kbytes through the DMA Configuration register, with the default being 128 bytes.

The start location for each receive AHB buffer is stored in memory in a list of receive buffer descriptors at an address location pointed to by the receive buffer queue pointer. The base address for the receive buffer queue pointer is configured in software using the Receive Buffer Queue Base Address register.

Each list entry consists of two words. The first is the address of the receive AHB buffer and the second the receive status. If the length of a receive frame exceeds the AHB buffer length, the status word for the used buffer is written with zeroes except for the “start of frame” bit, which is always set for the first buffer in a frame. Bit zero of the address field is written to 1 to show the buffer has been used. The receive buffer manager then reads the location of the next receive AHB buffer and fills that with the next part of the received frame data. AHB buffers are filled until the frame is complete and the final buffer descriptor status word contains the complete frame status. Refer to the table below for details of the receive buffer descriptor list.

Each receive AHB buffer start location is a word address. The start of the first AHB buffer in a frame can be offset by up to three bytes, depending on the value written to bits 14 and 15 of the Network Configuration register. If the start location of the AHB buffer is offset, the available length of the first AHB buffer is reduced by the corresponding number of bytes.

**Table 39-2. Receive Buffer Descriptor Entry**

| Bit | Function |
|---|---|
| Word 0 | |
| 31:2 | Address of beginning of buffer |
| 1 | Wrap—marks last descriptor in receive buffer descriptor list. |
| 0 | Ownership—needs to be zero for the GEMAC to write data to the receive buffer. The GEMAC sets this to one once it has successfully written a frame to memory.<br>Software has to clear this bit before the buffer can be used again. |
| Word 1 | |
| 31 | Global all ones broadcast address detected |
| 30 | Multicast hash match |
| 29 | Unicast hash match |
| 28 | – |

| .....continued | |
|---|---|
| **Bit** | **Function** |
| 27 | Specific Address Register match found, bit 25 and bit 26 indicate which Specific Address Register causes the match. |
| 26:25 | Specific Address Register match. Encoded as follows:<br>00: Specific Address Register 1 match<br>01: Specific Address Register 2 match<br>10: Specific Address Register 3 match<br>11: Specific Address Register 4 match<br>If more than one specific address is matched only one is indicated with priority 4 down to 1. |
| 24 | This bit has a different meaning depending on whether RX checksum offloading is enabled.<br>**With RX checksum offloading disabled:** (bit 24 clear in Network Configuration Register)<br>Type ID register match found, bit 22 and bit 23 indicate which type ID register causes the match.<br>**With RX checksum offloading enabled:** (bit 24 set in Network Configuration Register)<br>0: The frame was not SNAP encoded and/or had a VLAN tag with the Canonical Format Indicator (CFI) bit set.<br>1: The frame was SNAP encoded and had either no VLAN tag or a VLAN tag with the CFI bit not set. |
| 23:22 | This bit has a different meaning depending on whether RX checksum offloading is enabled.<br>**With RX checksum offloading disabled:** (bit 24 clear in Network Configuration)<br>Type ID register match. Encoded as follows:<br>00: Type ID register 1 match<br>01: Type ID register 2 match<br>10: Type ID register 3 match<br>11: Type ID register 4 match<br>If more than one Type ID is matched only one is indicated with priority 4 down to 1.<br>**With RX checksum offloading enabled:** (bit 24 set in Network Configuration Register)<br>00: Neither the IP header checksum nor the TCP/UDP checksum was checked.<br>01: The IP header checksum was checked and was correct. Neither the TCP nor UDP checksum was checked.<br>10: Both the IP header and TCP checksum were checked and were correct.<br>11: Both the IP header and UDP checksum were checked and were correct. |
| 21 | VLAN tag detected—type ID of 0x8100. For packets incorporating the stacked VLAN processing feature, this bit will be set if the second VLAN tag has a type ID of 0x8100 |

| .....continued | |
|---|---|
| **Bit** | **Function** |
| 20 | Priority tag detected—type ID of 0x8100 and null VLAN identifier. For packets incorporating the stacked VLAN processing feature, this bit will be set if the second VLAN tag has a type ID of 0x8100 and a null VLAN identifier. |
| 19:17 | VLAN priority—only valid if bit 21 is set. |
| 16 | Canonical format indicator (CFI) bit (only valid if bit 21 is set). |
| 15 | End of frame—when set the buffer contains the end of a frame. If end of frame is not set, then the only valid status bit is start of frame (bit 14). |
| 14 | Start of frame—when set the buffer contains the start of a frame. If both bits 15 and 14 are set, the buffer contains a whole frame. |
| 13 | This bit has a different meaning depending on whether jumbo frames and ignore FCS modes are enabled. If neither mode is enabled this bit will be zero.<br>**With jumbo frame mode enabled:** (bit 3 set in Network Configuration Register) Additional bit for length of frame (bit[13]), that is concatenated with bits[12:0]<br>**With ignore FCS mode enabled and jumbo frames disabled:** (bit 26 set in Network Configuration Register and bit 3 clear in Network Configuration Register) This indicates per frame FCS status as follows:<br>0: Frame had good FCS<br>1: Frame had bad FCS, but was copied to memory as ignore FCS enabled. |
| 12:0 | These bits represent the length of the received frame which may or may not include FCS depending on whether FCS discard mode is enabled.<br>**With FCS discard mode disabled:** (bit 17 clear in Network Configuration Register)<br>Least significant 12 bits for length of frame including FCS. If jumbo frames are enabled, these 12 bits are concatenated with bit[13] of the descriptor above.<br>**With FCS discard mode enabled:** (bit 17 set in Network Configuration Register)<br>Least significant 12 bits for length of frame excluding FCS. If jumbo frames are enabled, these 12 bits are concatenated with bit[13] of the descriptor above. |

To receive frames, the AHB buffer descriptors must be initialized by writing an appropriate address to bits 31:2 in the first word of each list entry. Bit 0 must be written with zero. Bit 1 is the wrap bit and indicates the last entry in the buffer descriptor list.

The start location of the receive buffer descriptor list must be written with the receive buffer queue base address before reception is enabled (receive enable in the Network Control register). Once reception is enabled, any writes to the Receive Buffer Queue Base Address register are ignored. When read, it will return the current pointer position in the descriptor list, though this is only valid and stable when receive is disabled.

If the filter block indicates that a frame should be copied to memory, the receive data DMA operation starts writing data into the receive buffer. If an error occurs, the buffer is recovered.

An internal counter within the GEMAC represents the receive buffer queue pointer and it is not visible through the CPU interface. The receive buffer queue pointer increments by two words after each buffer

has been used. It re-initializes to the receive buffer queue base address if any descriptor has its wrap bit set.

As receive AHB buffers are used, the receive AHB buffer manager sets bit zero of the first word of the descriptor to logic one indicating the AHB buffer has been used.

Software should search through the “used” bits in the AHB buffer descriptors to find out how many frames have been received, checking the start of frame and end of frame bits.

When the DMA is configured in the packet buffer Partial Store And Forward mode, received frames are written out to the AHB buffers as soon as enough frame data exists in the packet buffer. For both cases, this may mean several full AHB buffers are used before some error conditions can be detected. If a receive error is detected the receive buffer currently being written will be recovered. Previous buffers will not be recovered. As an example, when receiving frames with cyclic redundancy check (CRC) errors or excessive length, it is possible that a frame fragment might be stored in a sequence of AHB receive buffers. Software can detect this by looking for start of frame bit set in a buffer following a buffer with no end of frame bit set.

To function properly, a 10/100/1000 Ethernet system should have no excessive length frames or frames greater than 128 bytes with CRC errors. Collision fragments will be less than 128 bytes long, therefore it will be a rare occurrence to find a frame fragment in a receive AHB buffer, when using the default value of 128 bytes for the receive buffers size.

When in packet buffer full store and forward mode, only good received frames are written out of the DMA, so no fragments will exist in the AHB buffers due to MAC receiver errors. There is still the possibility of fragments due to DMA errors, for example used bit read on the second buffer of a multi-buffer frame.

If bit zero of the receive buffer descriptor is already set when the receive buffer manager reads the location of the receive AHB buffer, then the buffer has been already used and cannot be used again until software has processed the frame and cleared bit zero. In this case, the “buffer not available” bit in the Receive Status register is set and an interrupt triggered. The Receive Resource Error statistics register is also incremented.

When the DMA is configured in the packet buffer full store and forward mode, the user can optionally select whether received frames should be automatically discarded when no AHB buffer resource is available. This feature is selected via bit 24 of the DMA Configuration register (by default, the received frames are not automatically discarded). If this feature is off, then received packets will remain to be stored in the SRAM-based packet buffer until AHB buffer resource next becomes available. This may lead to an eventual packet buffer overflow if packets continue to be received when bit zero (used bit) of the receive buffer descriptor remains set. Note that after a used bit has been read, the receive buffer manager will re-read the location of the receive buffer descriptor every time a new packet is received. When the DMA is not configured in the packet buffer full store and forward mode and a used bit is read, the frame currently being received will be automatically discarded.

When the DMA is configured in the packet buffer full store and forward mode, a receive overrun condition occurs when the receive SRAM-based packet buffer is full, or because HRESP was not OK. In all other modes, a receive overrun condition occurs when either the AHB bus was not granted quickly enough, or because HRESP was not OK, or because a new frame has been detected by the receive block, but the status update or write back for the previous frame has not yet finished. For a receive overrun condition, the receive overrun interrupt is asserted and the buffer currently being written is recovered. The next frame that is received whose address is recognized reuses the buffer.

In any packet buffer mode, a write to bit 18 of GMAC_NCR will force a packet from the external SRAM-based receive packet buffer to be flushed. This feature is only acted upon when the RX DMA is not

currently writing packet data out to AHB, i.e., it is in an IDLE state. If the RX DMA is active, a write to this bit is ignored.

#### 39.6.3.4 Transmit AHB Buffers

Frames to transmit are stored in one or more transmit AHB buffers. Transmit frames can be between 1 and 16384 bytes long, so it is possible to transmit frames longer than the maximum length specified in the IEEE 802.3 standard. It should be noted that zero length AHB buffers are allowed and that the maximum number of buffers permitted for each transmit frame is 128.

The start location for each transmit AHB buffer is stored in memory in a list of transmit buffer descriptors at a location pointed to by the transmit buffer queue pointer. The base address for this queue pointer is set in software using the Transmit Buffer Queue Base Address register. Each list entry consists of two words. The first is the byte address of the transmit buffer and the second containing the transmit control and status. For the packet buffer DMA, the start location for each AHB buffer is a byte address, the bottom bits of the address being used to offset the start of the data from the data-word boundary (i.e., bits 2,1 and 0 are used to offset the address for 64-bit datapaths).

Frames can be transmitted with or without automatic CRC generation. If CRC is automatically generated, pad will also be automatically generated to take frames to a minimum length of 64 bytes. When CRC is not automatically generated (as defined in word 1 of the transmit buffer descriptor), the frame is assumed to be at least 64 bytes long and pad is not generated.

An entry in the transmit buffer descriptor list is described in the table below.

To transmit frames, the buffer descriptors must be initialized by writing an appropriate byte address to bits [31:0] in the first word of each descriptor list entry.

The second word of the transmit buffer descriptor is initialized with control information that indicates the length of the frame, whether or not the MAC is to append CRC and whether the buffer is the last buffer in the frame.

After transmission the status bits are written back to the second word of the first buffer along with the used bit. Bit 31 is the used bit which must be zero when the control word is read if transmission is to take place. It is written to one once the frame has been transmitted. Bits[29:20] indicate various transmit error conditions. Bit 30 is the wrap bit which can be set for any buffer within a frame. If no wrap bit is encountered the queue pointer continues to increment.

The Transmit Buffer Queue Base Address register can only be updated while transmission is disabled or halted; otherwise any attempted write will be ignored. When transmission is halted the transmit buffer queue pointer will maintain its value. Therefore when transmission is restarted the next descriptor read from the queue will be from immediately after the last successfully transmitted frame. while transmit is disabled (bit 3 of the Network Control register set low), the transmit buffer queue pointer resets to point to the address indicated by the Transmit Buffer Queue Base Address register. Note that disabling receive does not have the same effect on the receive buffer queue pointer.

Once the transmit queue is initialized, transmit is activated by writing to the transmit start bit (bit 9) of the Network Control register. Transmit is halted when a buffer descriptor with its used bit set is read, a transmit error occurs, or by writing to the transmit halt bit of the Network Control register. Transmission is suspended if a pause frame is received while the pause enable bit is set in the Network Configuration register. Rewriting the start bit while transmission is active is allowed. This is implemented with TXGO variable which is readable in the Transmit Status register at bit location 3. The TXGO variable is reset when:

- Transmit is disabled.
- A buffer descriptor with its ownership bit set is read.

- Bit 10, THALT, of the Network Control register is written.
- There is a transmit error such as too many retries or a transmit underrun.

To set TXGO, write TSTART to the bit 9 of the Network Control register. Transmit halt does not take effect until any ongoing transmit finishes.

If the DMA is configured for packet buffer Partial Store and Forward mode and a collision occurs during transmission of a multi-buffer frame, transmission will automatically restart from the first buffer of the frame. For packet buffer mode, the entire contents of the frame are read into the transmit packet buffer memory, so the retry attempt will be replayed directly from the packet buffer memory rather than having to re-fetch through the AHB.

If a used bit is read midway through transmission of a multi-buffer frame, this is treated as a transmit error. Transmission stops, GTXER is asserted and the FCS will be bad.

If transmission stops due to a transmit error or a used bit being read, transmission restarts from the first buffer descriptor of the frame being transmitted when the transmit start bit is rewritten.

**Table 39-3. Transmit Buffer Descriptor Entry**

| Bit | Function |
|---|---|
| Word 0 | |
| 31:0 | Byte address of buffer |
| Word 1 | |
| 31 | Used—must be zero for the GEMAC to read data to the transmit buffer. The GEMAC sets this to one for the first buffer of a frame once it has been successfully transmitted. Software must clear this bit before the buffer can be used again. |
| 30 | Wrap—marks last descriptor in transmit buffer descriptor list. This can be set for any buffer within the frame. |
| 29 | Retry limit exceeded, transmit error detected |
| 28 | Reserved. |
| 27 | Transmit frame corruption due to AHB error—set if an error occurs while midway through reading transmit frame from the AHB, including HRESP errors and buffers exhausted mid frame (if the buffers run out during transmission of a frame then transmission stops, FCS shall be bad and GTXER asserted).<br>Also set if single frame is too large for configured packet buffer memory size. |
| 26 | Late collision, transmit error detected. |
| 25:23 | Reserved |

| .....continued | |
|---|---|
| **Bit** | **Function** |
| 22:20 | Transmit IP/TCP/UDP checksum generation offload errors:<br>000: No Error.<br>001: The Packet was identified as a VLAN type, but the header was not fully complete, or had an error in it.<br>010: The Packet was identified as a SNAP type, but the header was not fully complete, or had an error in it.<br>011: The Packet was not of an IP type, or the IP packet was invalidly short, or the IP was not of type IPv4/IPv6.<br>100: The Packet was not identified as VLAN, SNAP or IP.<br>101: Non supported packet fragmentation occurred. For IPv4 packets, the IP checksum was generated and inserted.<br>110: Packet type detected was not TCP or UDP. TCP/UDP checksum was therefore not generated. For IPv4 packets, the IP checksum was generated and inserted.<br>111: A premature end of packet was detected and the TCP/UDP checksum could not be generated. |
| 19:17 | Reserved |
| 16 | No CRC to be appended by MAC. When set, this implies that the data in the buffers already contains a valid CRC, hence no CRC or padding is to be appended to the current frame by the MAC.<br>This control bit must be set for the first buffer in a frame and will be ignored for the subsequent buffers of a frame.<br>Note that this bit must be clear when using the transmit IP/TCP/UDP checksum generation offload, otherwise checksum generation and substitution will not occur. |
| 15 | Last buffer, when set this bit will indicate the last buffer in the current frame has been reached. |
| 14 | Reserved |
| 13:0 | Length of buffer |

#### 39.6.3.5 DMA Bursting on the AHB

The DMA will always use SINGLE, or INCR type AHB accesses for buffer management operations. When performing data transfers, the AHB burst length used can be programmed using bits 4:0 of the DMA Configuration register so that either SINGLE, INCR or fixed length incrementing bursts (INCR4, INCR8 or INCR16) are used where possible.

When there is enough space and enough data to be transferred, the programmed fixed length bursts will be used. If there is not enough data or space available, for example when at the beginning or the end of a buffer, SINGLE type accesses are used. Also SINGLE type accesses are used at 1024 byte boundaries, so that the 1 Kbyte boundaries are not burst over as per AHB requirements.

The DMA will not terminate a fixed length burst early, unless an error condition occurs on the AHB or if receive or transmit are disabled in the Network Control register.

#### 39.6.3.6 DMA Packet Buffer

The DMA uses packet buffers for both transmit and receive paths. This mode allows multiple packets to be buffered in both transmit and receive directions. This allows the DMA to withstand far greater access latencies on the AHB and make more efficient use of the AHB bandwidth. There are two modes of operation—Full Store and Forward and Partial Store and Forward.

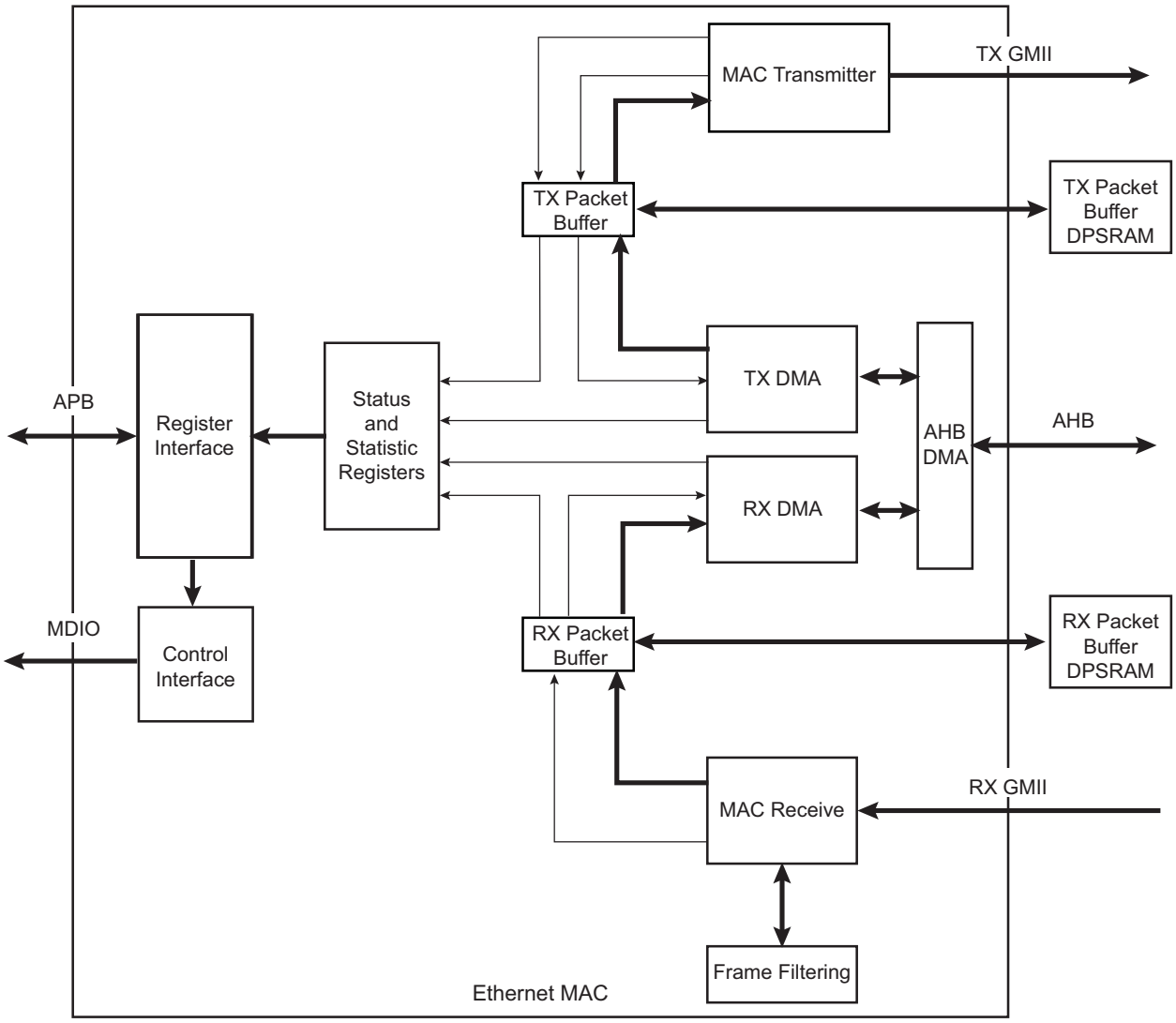
As described in section Partial Store and Forward Using Packet Buffer DMA, the DMA can be programmed into a low latency mode, known as Partial Store and Forward.

When the DMA is in full store and forward mode, full packets are buffered which provides the possibility to:

- Discard packets with error on the receive path before they are partially written out of the DMA, thus saving AHB bus bandwidth and driver processing overhead,
- Retry collided transmit frames from the buffer, thus saving AHB bus bandwidth,
- Implement transmit IP/TCP/UDP checksum generation offload.

With the packet buffers included, the structure of the GEMAC data paths is shown in the figure below.

**Figure 39-2. Data Paths with Packet Buffers Included**



#### 39.6.3.7 Transmit Packet Buffer

The transmitter packet buffer will continue attempting to fetch frame data from the AHB system memory until the packet buffer itself is full, at which point it will attempt to maintain its full level.

To accommodate the status and statistics associated with each frame, three words per packet are reserved at the end of the packet data. If the packet is bad and requires to be dropped, the status and statistics are the only information held on that packet. Storing the status in the DPRAM is required in order to decouple the DMA interface of the buffer from the MAC interface, to update the MAC status/ statistics and to generate interrupts in the order in which the packets that they represent were fetched from the AHB memory.

If any errors occur on the AHB while reading the transmit frame, the fetching of packet data from AHB memory is halted. The MAC transmitter will continue to fetch packet data, thereby emptying the packet buffer and allowing any good non-errored frames to be transmitted successfully. Once these have been fully transmitted, the status/statistics for the errored frame will be updated and software will be informed via an interrupt that an AHB error occurred. This way, the error is reported in the correct packet order.

The transmit packet buffer will only attempt to read more frame data from the AHB when space is available in the packet buffer memory. If space is not available it must wait until the a packet fetched by the MAC completes transmission and is subsequently removed from the packet buffer memory. Note that if full store and forward mode is active and if a single frame is fetched that is too large for the packet buffer memory, the frame is flushed and the DMA halted with an error status. This is because a complete frame must be written into the packet buffer before transmission can begin, and therefore the minimum packet buffer memory size should be chosen to satisfy the maximum frame to be transmitted in the application.

In full store and forward mode, once the complete transmit frame is written into the packet buffer memory, a trigger is sent across to the MAC transmitter, which will then begin reading the frame from the packet buffer memory. Since the whole frame is present and stable in the packet buffer memory an underflow of the transmitter is not possible. The frame is kept in the packet buffer until notification is received from the MAC that the frame data has either been successfully transmitted or can no longer be retransmitted (too many retries in half duplex mode). When this notification is received the frame is flushed from memory to make room for a new frame to be fetched from AHB system memory.

In Partial Store and Forward mode, a trigger is sent across to the MAC transmitter as soon as sufficient packet data is available, which will then begin fetching the frame from the packet buffer memory. If, after this point, the MAC transmitter is able to fetch data from the packet buffer faster than the AHB DMA can fill it, an underflow of the transmitter is possible. In this case, the transmission is terminated early, and the packet buffer is completely flushed. Transmission can only be restarted by writing to the transmit START bit.

In half duplex mode, the frame is kept in the packet buffer until notification is received from the MAC that the frame data has either been successfully transmitted or can no longer be retransmitted (too many retries in half duplex mode). When this notification is received the frame is flushed from memory to make room for a new frame to be fetched from AHB system memory.

In full duplex mode, the frame is removed from the packet buffer on the fly.

Other than underflow, the only MAC related errors that can occur are due to collisions during half duplex transmissions. When a collision occurs the frame still exists in the packet buffer memory so can be retried directly from there. Only once the MAC transmitter has failed to transmit after sixteen attempts is the frame finally flushed from the packet buffer.

#### 39.6.3.8 Receive Packet Buffer

The receive packet buffer stores frames from the MAC receiver along with their status and statistics. Frames with errors are flushed from the packet buffer memory, while good frames are pushed onto the DMA AHB interface.

The receiver packet buffer monitors the FIFO write interface from the MAC receiver and translates the FIFO pushes into packet buffer writes. At the end of the received frame the status and statistics are buffered so that the information can be used when the frame is read out. When programmed in full store and forward mode, if the frame has an error the frame data is immediately flushed from the packet buffer memory allowing subsequent frames to utilise the freed up space. The status and statistics for bad frames are still used to update the GEMAC registers.

To accommodate the status and statistics associated with each frame, three words per packet (or two if configured in 64-bit datapath mode) are reserved at the end of the packet data. If the packet is bad and requires to be dropped, the status and statistics are the only information held on that packet.

The receiver packet buffer will also detect a full condition so that an overflow condition can be detected. If this occurs, subsequent packets are dropped and an RX overflow interrupt is raised.

For full store and forward, the DMA only begins packet fetches once the status and statistics for a frame are available. If the frame has a bad status due to a frame error, the status and statistics are passed on to the GEMAC registers. If the frame has a good status, the information is used to read the frame from the packet buffer memory and burst onto the AHB using the DMA buffer management protocol. Once the last frame data has been transferred to the packet buffer, the status and statistics are updated to the GEMAC registers.

If Partial Store and Forward mode is active, the DMA will begin fetching the packet data before the status is available. As soon as the status becomes available, the DMA will fetch this information as soon as possible before continuing to fetch the remainder of the frame. Once the last frame data has been transferred to the packet buffer, the status and statistics are updated to the GEMAC registers.

#### 39.6.3.9 Priority Queueing in the DMA

The DMA by default uses a single transmit and receive queue. This means the list of transmit/receive buffer descriptors point to data buffers associated with a single transmit/receive data stream. The GEMAC can select up to 3 priority queues. Each queue has an independent list of buffer descriptors pointing to separate data streams.

The table below gives the DPRAM size associated with each queue:

**Table 39-4. Queue Size**

| Queue Number | Queue Size |
|---|---|
| 2 | 4 KB |
| 1 | 2 KB |
| 0 (lowest priority) | 2 KB |

In the transmit direction, higher priority queues are always serviced before lower priority queues, with Q0 as lowest priority and Q2 as highest priority. This strict priority scheme requires the user to ensure that high priority traffic is constrained so that lower priority traffic will have required bandwidth. The GEMAC DMA will determine the next queue to service by initiating a sequence of buffer descriptor reads interrogating the ownership bits of each. The buffer descriptor corresponding to the highest priority queue is read first. As an example, if the ownership bit of this descriptor is set, then the DMA will progress to

reading the 2nd highest priority queue's descriptor. If that ownership bit read of this lower priority queue is set, then the DMA will read the 3rd highest priority queue's descriptor. If all the descriptors return an ownership bit set, then a resource error has occurred, an interrupt is generated and transmission is automatically halted. Transmission can only be restarted by setting the START bit in the Network Control register. The GEMAC DMA will need to identify the highest available queue to transmit from when the START bit in the Network Control register is written to and the TX is in a halted state, or when the last word of any packet has been fetched from external AHB memory.

The GEMAC transmit DMA maximizes the effectiveness of priority queuing by ensuring that high priority traffic be transmitted as early as possible after being fetched from AHB. High priority traffic fetched from AHB will be pushed to the MAC layer, depending on traffic shaping being enabled and the associated credit value for that queue, before any lower priority traffic that may pre-exist in the transmit SRAM-based packet buffer. This is achieved by separating the transmit SRAM-based packet buffer into regions, one region per queue. The size of each region determines the amount of SRAM space allocated per queue.

For each queue, there is an associated Transmit Buffer Queue Base Address register. For the lowest priority queue (or the only queue when only one queue is selected), the Transmit Buffer Queue Base Address is located at address 0x1C. For all other queues, the Transmit Buffer Queue Base Address registers are located at sequential addresses starting at address 0x440.

In the receive direction each packet is written to AHB data buffers in the order that it is received. For each queue, there is an independent set of receive AHB buffers for each queue. There is therefore a separate Receive Buffer Queue Base Address register for each queue. For the lowest priority queue (or the only queue when only one queue is selected), the Receive Buffer Queue Base Address is located at address 0x18. For all other queues, the Receive Buffer Queue Base Address registers are located at sequential addresses starting at address 0x480. Every received packet will pass through a programmable screening algorithm which will allocate a particular queue to that frame. The user interface to the screeners is through two types of programmable registers:

- Screening Type 1 registers—The module features 4 Screening Type 1 registers. Screening Type 1 registers hold values to match against specific IP and UDP fields of the received frames. The fields matched against are DS (Differentiated Services field of IPv4 frames), TC (Traffic class field of IPv6 frames) and/or the UDP destination port.
- Screening Type 2 registers—The module features 8 Screening Type 2 registers GMAC_ST2RPQ. Screening Type 2 registers operate independently of Screening Type 1 registers and offer additional match capabilities. Screening Type 2 allows a screen to be configured that is the combination of all or any of the following comparisons:

1. An enable bit VLAN priority, VLANE. A VLAN priority match will be performed if the VLAN priority enable is set. The extracted priority field in the VLAN header is compared against VLANP in the GMAC_ST2RPQ register itself.
2. An enable bit EtherType, ETHE. The EtherType field I2ETH inside GMAC_ST2RPQ maps to one of 4 EtherType match registers, GMAC_ST2ER. The extracted EtherType is compared against GMAC_ST2ER designated by this EtherType field.
3. An enable bit Compare A, COMPAE. This bit is associated with a Screening Type 2 Compare Word 0/1 register x, GMAC_ST2CW0/1.
4. An enable bit Compare B, COMPBE. This bit is associated with a Screening Type 2 Compare Word 0/1 register x, GMAC_ST2CW0/1.
5. An enable bit Compare C, COMPCE. This bit is associated with a Screening Type 2 Compare Word 0/1 register x, GMAC_ST2CW0/1.

Each screener type has an enable bit, a match pattern and a queue number. If a received frame matches on an enabled Screening register, then the frame will be tagged with the queue value in the associated Screening register, and forwarded onto the DMA and subsequently into the external memory associated with that queue. If two screeners are matched, then the one which resides at the lowest register address will take priority so care must be taken on the selection of the screener location.

When the priority queuing feature is enabled, the number of interrupt outputs from the GEMAC core is increased to match the number of supported queues. The number of Interrupt Status registers is increased by the same number. Only DMA related events are reported using the individual interrupt outputs, as the GEMAC can relate these events to specific queues. All other events generated within the GEMAC are reported in the interrupt associated with the lowest priority queue. For the lowest priority queue (or the only queue when only 1 queue is selected), the Interrupt Status register is located at address 0x24. For all other queues, the Interrupt Status register is located at sequential addresses starting at address 0x400.

**Note:** The address matching is the first level of filtering. If there is a match, the screeners are the next level of filtering for routing the data to the appropriate queue. See MAC Filtering Block for more details.

The additional screening done by the functions Compare A, B, and C each have an enable bit and compare register field. COMPA, COMPB and COMPC in GMAC_ST2RPQ are pointers to a configured offset (OFFSVAL), value (COMPVAL), and mask (MASKVAL). If enabled, the compare is true if the data at the offset into the frame, ANDed with MASKVAL, is equal to the value of COMPVAL ANDed with MASKVAL. A 16-bit word comparison is done. The byte at the offset number of bytes from the index start is compared to bits 7:0 of the configured COMPVAL and MASKVAL. The byte at the offset number of bytes + 1 from the index start is compared to bits 15:8 of the configured COMPVAL and MASKVAL.

The offset value in bytes, OFFSVAL, ranges from 0 to 127 bytes from either the start of the frame, the byte after the EtherType field, the byte after the IP header (IPv4 or IPv6) or the byte after the TCP/UDP header. Note the logic to decode the IP header or the TCP/UDP header is reused from the TCP/UDP/IP checksum offload logic and therefore has the same restrictions on use (the main limitation is that IP fragmentation is not supported). Refer to the Checksum Offload for IP, TCP and UDP section of this documentation for further details.

Compare A, B, and C use a common set of 24 GMAC_ST2CW0/1 registers, thus all COMPA, COMPB and COMPC fields in the registers GMAC_ST2RPQ point to a single pool of 24 GMAC_ST2CW0/1 registers.

Note that Compare A, B and C together allow matching against an arbitrary 48 bits of data and so can be used to match against a MAC address.

All enabled comparisons are ANDed together to form the overall type 2 screening match.

### 39.6.4 MAC Transmit Block

The MAC transmitter can operate in either half duplex or full duplex mode and transmits frames in accordance with the Ethernet IEEE 802.3 standard. In half duplex mode, the CSMA/CD protocol of the IEEE 802.3 specification is followed.

A small input buffer receives data through the FIFO interface which will extract data in 32-bit form. All subsequent processing prior to the final output is performed in bytes.

Transmit data can be output using the MII interface.

Frame assembly starts by adding preamble and the start frame delimiter. Data is taken from the transmit FIFO interface a word at a time.

If necessary, padding is added to take the frame length to 60 bytes. CRC is calculated using an order 32-bit polynomial. This is inverted and appended to the end of the frame taking the frame length to a minimum of 64 bytes. If the no CRC bit is set in the second word of the last buffer descriptor of a transmit frame, neither pad nor CRC are appended. The no CRC bit can also be set through the FIFO interface.

In full duplex mode (at all data rates), frames are transmitted immediately. Back to back frames are transmitted at least 96 bit times apart to guarantee the interframe gap.

In half duplex mode, the transmitter checks carrier sense. If asserted, the transmitter waits for the signal to become inactive, and then starts transmission after the interframe gap of 96 bit times. If the collision signal is asserted during transmission, the transmitter will transmit a jam sequence of 32 bits taken from the data register and then retry transmission after the backoff time has elapsed. If the collision occurs during either the preamble or Start Frame Delimiter (SFD), then these fields will be completed prior to generation of the jam sequence.

The backoff time is based on an XOR of the 10 least significant bits of the data coming from the transmit FIFO interface and a 10-bit pseudo random number generator. The number of bits used depends on the number of collisions seen. After the first collision 1 bit is used, then the second 2 bits and so on up to the maximum of 10 bits. All 10 bits are used above ten collisions. An error will be indicated and no further attempts will be made if 16 consecutive attempts cause collision. This operation is compliant with the description in Clause 4.2.3.2.5 of the IEEE 802.3 standard which refers to the truncated binary exponential backoff algorithm.

In 10/100 mode, both collisions and late collisions are treated identically, and backoff and retry will be performed up to 16 times. This condition is reported in the transmit buffer descriptor word 1 (late collision, bit 26) and also in the Transmit Status register (late collision, bit 7). An interrupt can also be generated (if enabled) when this exception occurs, and bit 5 in the Interrupt Status register will be set.

In all modes of operation, if the transmit DMA underruns, a bad CRC is automatically appended using the same mechanism as jam insertion and the GTXER signal is asserted. For a properly configured system this should never happen and also it is impossible if configured to use the DMA with packet buffers, as the complete frame is buffered in local packet buffer memory.

By setting when bit 28 is set in the Network Configuration register, the Inter Packet Gap (IPG) may be stretched beyond 96 bits depending on the length of the previously transmitted frame and the value written to the IPG Stretch register (GMAC_IPGS). The least significant 8 bits of the IPG Stretch register multiply the previous frame length (including preamble). The next significant 8 bits (+1 so as not to get a divide by zero) divide the frame length to generate the IPG. IPG stretch only works in full duplex mode and when bit 28 is set in the Network Configuration register. The IPG Stretch register cannot be used to shrink the IPG below 96 bits.

### 39.6.5 MAC Receive Block

All processing within the MAC receive block is implemented using a 16-bit data path. The MAC receive block checks for valid preamble, FCS, alignment and length, presents received frames to the FIFO interface and stores the frame destination address for use by the address checking block.

If, during the frame reception, the frame is found to be too long, a bad frame indication is sent to the FIFO interface. The receiver logic ceases to send data to memory as soon as this condition occurs.

At end of frame reception the receive block indicates to the DMA block whether the frame is good or bad. The DMA block will recover the current receive buffer if the frame was bad.

Ethernet frames are normally stored in DMA memory complete with the FCS. Setting the FCS remove bit in the network configuration (bit 17) causes frames to be stored without their corresponding FCS. The reported frame length field is reduced by four bytes to reflect this operation.

The receive block signals to the register block to increment the alignment, CRC (FCS), short frame, long frame, jabber or receive symbol errors when any of these exception conditions occur.

If bit 26 is set in the network configuration, CRC errors will be ignored and CRC errored frames will not be discarded, though the Frame Check Sequence Errors statistic register will still be incremented. Additionally, if not enabled for jumbo frames mode, then bit[13] of the receiver descriptor word 1 will be updated to indicate the FCS validity for the particular frame. This is useful for applications such as EtherCAT whereby individual frames with FCS errors must be identified.

Received frames can be checked for length field error by setting the length field error frame discard bit of the Network Configuration register (bit-16). When this bit is set, the receiver compares a frame's measured length with the length field (bytes 13 and 14) extracted from the frame. The frame is discarded if the measured length is shorter. This checking procedure is for received frames between 64 bytes and 1518 bytes in length.

Each discarded frame is counted in the 10-bit Length Field Frame Error statistics register. Frames where the length field is greater than or equal to 0x0600 hex will not be checked.

### 39.6.6 Checksum Offload for IP, TCP and UDP

The GEMAC can be programmed to perform IP, TCP and UDP checksum offloading in both receive and transmit directions, which is enabled by setting bit 24 in the Network Configuration register for receive and bit 11 in the DMA Configuration register for transmit.

IPv4 packets contain a 16-bit checksum field, which is the 16-bit 1’s complement of the 1’s complement sum of all 16-bit words in the header. TCP and UDP packets contain a 16-bit checksum field, which is the 16-bit 1’s complement of the 1’s complement sum of all 16-bit words in the header, the data and a conceptual IP pseudo header.

To calculate these checksums in software requires each byte of the packet to be processed. For TCP and UDP this can use a large amount of processing power. Offloading the checksum calculation to hardware can result in significant performance improvements.

For IP, TCP or UDP checksum offload to be useful, the operating system containing the protocol stack must be aware that this offload is available so that it can make use of the fact that the hardware can either generate or verify the checksum.

#### 39.6.6.1 Receiver Checksum Offload

When receive checksum offloading is enabled in the GEMAC, the IPv4 header checksum is checked as per RFC 791, where the packet meets the following criteria:

- If present, the VLAN header must be four octets long and the CFI bit must not be set.
- Encapsulation must be RFC 894 Ethernet Type Encoding or RFC 1042 SNAP Encoding.
- IPv4 packet
- IP header is of a valid length

The GEMAC also checks the TCP checksum as per RFC 793, or the UDP checksum as per RFC 768, if the following criteria are met:

- IPv4 or IPv6 packet
- Good IP header checksum (if IPv4)

- No IP fragmentation
- TCP or UDP packet

When an IP, TCP or UDP frame is received, the receive buffer descriptor gives an indication if the GEMAC was able to verify the checksums. There is also an indication if the frame had SNAP encapsulation. These indication bits will replace the type ID match indication bits when the receive checksum offload is enabled. For details of these indication bits, refer to Receive Buffer Descriptor Entry.

If any of the checksums are verified as incorrect by the GEMAC, the packet is discarded and the appropriate statistics counter incremented.

#### 39.6.6.2 Transmitter Checksum Offload

The transmitter checksum offload is only available if the full store and forward mode is enabled. This is because the complete frame to be transmitted must be read into the packet buffer memory before the checksum can be calculated and written back into the headers at the beginning of the frame.

Transmitter checksum offload is enabled by setting bit [11] in the DMA Configuration register. When enabled, it will monitor the frame as it is written into the transmitter packet buffer memory to automatically detect the protocol of the frame. Protocol support is identical to the receiver checksum offload.

For transmit checksum generation and substitution to occur, the protocol of the frame must be recognized and the frame must be provided without the FCS field, by making sure that bit [16] of the transmit descriptor word 1 is clear. If the frame data already had the FCS field, this would be corrupted by the substitution of the new checksum fields.

If these conditions are met, the transmit checksum offload engine will calculate the IP, TCP and UDP checksums as appropriate. Once the full packet is completely written into packet buffer memory, the checksums will be valid and the relevant DPRAM locations will be updated for the new checksum fields as per standard IP/TCP and UDP packet structures.

If the transmitter checksum engine is prevented from generating the relevant checksums, bits [22:20] of the transmitter DMA writeback status will be updated to identify the reason for the error. Note that the frame will still be transmitted but without the checksum substitution, as typically the reason that the substitution did not occur was that the protocol was not recognized.

### 39.6.7 MAC Filtering Block

The filter block determines which frames should be written to the FIFO interface and on to the DMA.

Whether a frame is passed depends on what is enabled in the Network Configuration register, the state of the external matching pins, the contents of the specific address, type and Hash registers and the frame's destination address and type field.

If bit 25 of the Network Configuration register is not set, a frame will not be copied to memory if the GEMAC is transmitting in half duplex mode at the time a destination address is received.

Ethernet frames are transmitted a byte at a time, least significant bit first. The first six bytes (48 bits) of an Ethernet frame make up the destination address. The first bit of the destination address, which is the LSB of the first byte of the frame, is the group or individual bit. This is one for multicast addresses and zero for unicast. The all ones address is the broadcast address and a special case of multicast.

The GEMAC supports recognition of four specific addresses. Each specific address requires two registers, Specific Address Bottom register and Specific Address Top register. Specific Address Bottom register stores the first four bytes of the destination address and Specific Address Top register contains the last two bytes. The addresses stored can be specific, group, local or universal.

The destination address of received frames is compared against the data stored in the Specific Address registers once they have been activated. The addresses are deactivated at reset or when their corresponding Specific Address Bottom register is written. They are activated when Specific Address Top register is written. If a receive frame address matches an active address, the frame is written to the FIFO interface and on to DMA memory.

Frames may be filtered using the type ID field for matching. Four type ID registers exist in the register address space and each can be enabled for matching by writing a one to the MSB (bit 31) of the respective register. When a frame is received, the matching is implemented as an OR function of the various types of match.

The contents of each type ID register (when enabled) are compared against the length/type ID of the frame being received (e.g., bytes 13 and 14 in non-VLAN and non-SNAP encapsulated frames) and copied to memory if a match is found. The encoded type ID match bits (Word 0, Bit 22 and Bit 23) in the receive buffer descriptor status are set indicating which type ID register generated the match, if the receive checksum offload is disabled.

The reset state of the type ID registers is zero, hence each is initially disabled.

The following example illustrates the use of the address and type ID match registers for a MAC address of 21:43:65:87:A9:CB:

| | |
|---|---|
| Preamble | 55 |
| SFD | D5 |
| DA (Octet 0 - LSB) | 21 |
| DA (Octet 1) | 43 |
| DA (Octet 2) | 65 |
| DA (Octet 3) | 87 |
| DA (Octet 4) | A9 |
| DA (Octet 5 - MSB) | CB |
| SA (LSB) | 00(1) |
| SA | 00(1) |
| SA | 00(1) |
| SA | 00(1) |
| SA | 00(1) |
| SA (MSB) | 00(1) |
| Type ID (MSB) | 43 |
| Type ID (LSB) | 21 |

**Note:**

1. Contains the address of the transmitting device.

The sequence above shows the beginning of an Ethernet frame. Byte order of transmission is from top to bottom as shown. For a successful match to specific address 1, the following address matching registers must be set up:

- Specific Address 1 Bottom register (GMAC_SAB1) (Address 0x088) 0x87654321
- Specific Address 1 Top register (GMAC_SAT1) (Address 0x08C) 0x0000CBA9

For a successful match to the type ID, the following Type ID Match 1 register must be set up:

- Type ID Match 1 register (GMAC_TIDM1) (Address 0x0A8) 0x80004321

### 39.6.8 Broadcast Address

Frames with the broadcast address of 0xFFFFFFFFFFFF are stored to memory only if the 'no broadcast' bit in the Network Configuration register is set to zero.

### 39.6.9 Hash Addressing

The hash address register is 64 bits long and takes up two locations in the memory map. The least significant bits are stored in Hash Register Bottom and the most significant bits in Hash Register Top.

The unicast hash enable and the multicast hash enable bits in the Network Configuration register enable the reception of hash matched frames. The destination address is reduced to a 6-bit index into the 64-bit Hash register using the following hash function: The hash function is an XOR of every sixth bit of the destination address.

```
hash_index[05] = da[05] ^ da[11] ^ da[17] ^ da[23] ^ da[29] ^ da[35] ^ da[41] ^ da[47]
hash_index[04] = da[04] ^ da[10] ^ da[16] ^ da[22] ^ da[28] ^ da[34] ^ da[40] ^ da[46]
hash_index[03] = da[03] ^ da[09] ^ da[15] ^ da[21] ^ da[27] ^ da[33] ^ da[39] ^ da[45]
hash_index[02] = da[02] ^ da[08] ^ da[14] ^ da[20] ^ da[26] ^ da[32] ^ da[38] ^ da[44]
hash_index[01] = da[01] ^ da[07] ^ da[13] ^ da[19] ^ da[25] ^ da[31] ^ da[37] ^ da[43]
hash_index[00] = da[00] ^ da[06] ^ da[12] ^ da[18] ^ da[24] ^ da[30] ^ da[36] ^ da[42]
da[0]
```

represents the least significant bit of the first byte received, that is, the multicast/unicast indicator, and `da[47]` represents the most significant bit of the last byte received.

If the hash index points to a bit that is set in the Hash register then the frame will be matched according to whether the frame is multicast or unicast.

A multicast match will be signalled if the multicast hash enable bit is set, `da[0]` is logic 1 and the hash index points to a bit set in the Hash register.

A unicast match will be signalled if the unicast hash enable bit is set, `da[0]` is logic 0 and the hash index points to a bit set in the Hash register.

To receive all multicast frames, the Hash register should be set with all ones and the multicast hash enable bit should be set in the Network Configuration register.

### 39.6.10 Copy all Frames (Promiscuous Mode)

If the Copy All Frames bit is set in the Network Configuration register then all frames except those that are too long, too short, have FCS errors or have GRXER asserted during reception will be copied to memory. Frames with FCS errors will be copied if bit 26 is set in the Network Configuration register.

### 39.6.11 Disable Copy of Pause Frames

Pause frames can be prevented from being written to memory by setting the disable copying of pause frames control bit 23 in the Network Configuration register. When set, pause frames are not copied to memory regardless of the Copy All Frames bit, whether a hash match is found, a type ID match is identified or if a destination address match is found.

### 39.6.12 VLAN Support

The following table describes an Ethernet encoded 802.1Q VLAN tag.

**Table 39-5. 802.1Q VLAN Tag**

| TPID (Tag Protocol Identifier) 16 bits | TCI (Tag Control Information) 16 bits |
|---|---|
| 0x8100 | First 3 bits priority, then CFI bit, last 12 bits VID |

The VLAN tag is inserted at the 13th byte of the frame adding an extra four bytes to the frame. To support these extra four bytes, the GEMAC can accept frame lengths up to 1536 bytes by setting bit 8 in the Network Configuration register.

If the VID (VLAN identifier) is null (0x000) this indicates a priority-tagged frame.

The following bits in the receive buffer descriptor status word give information about VLAN tagged frames:-

- Bit 21 set if receive frame is VLAN tagged (i.e., type ID of 0x8100).
- Bit 20 set if receive frame is priority tagged (i.e., type ID of 0x8100 and null VID). (If bit 20 is set, bit 21 will be set also.)
- Bit 19, 18 and 17 set to priority if bit 21 is set.
- Bit 16 set to CFI if bit 21 is set.

The GEMAC can be configured to reject all frames except VLAN tagged frames by setting the discard non-VLAN frames bit in the Network Configuration register.

### 39.6.13 Wake on LAN Support

The receive block supports Wake on LAN by detecting the following events on incoming receive frames:

- Magic packet
- Address Resolution Protocol (ARP) request to the device IP address
- Specific address 1 filter match
- Multicast hash filter match

These events can be individually enabled through bits [19:16] of the Wake on LAN register. Also, for Wake on LAN detection to occur, receive enable must be set in the Network Control register, however a receive buffer does not have to be available.

In case of an ARP request, specific address 1 or multicast filter events will occur even if the frame is errored. For magic packet events, the frame must be correctly formed and error free.

A magic packet event is detected if all of the following are true:

- Magic packet events are enabled through bit 16 of the Wake on LAN register
- The frame's destination address matches specific address 1
- The frame is correctly formed with no errors
- The frame contains at least 6 bytes of 0xFF for synchronization
- There are 16 repetitions of the contents of Specific Address 1 register immediately following the synchronization

An ARP request event is detected if all of the following are true:

- ARP request events are enabled through bit 17 of the Wake on LAN register
- Broadcasts are allowed by bit 5 in the Network Configuration register
- The frame has a broadcast destination address (bytes 1 to 6)
- The frame has a type ID field of 0x0806 (bytes 13 and 14)

- The frame has an ARP operation field of 0x0001 (bytes 21 and 22)
- The least significant 16 bits of the frame's ARP target protocol address (bytes 41 and 42) match the value programmed in bits[15:0] of the Wake on LAN register

The decoding of the ARP fields adjusts automatically if a VLAN tag is detected within the frame. The reserved value of 0x0000 for the Wake on LAN target address value will not cause an ARP request event, even if matched by the frame.

A specific address 1 filter match event will occur if all of the following are true:

- Specific address 1 events are enabled through bit 18 of the Wake on LAN register
- The frame's destination address matches the value programmed in the Specific Address 1 registers

A multicast filter match event will occur if all of the following are true:

- Multicast hash events are enabled through bit 19 of the Wake on LAN register
- Multicast hash filtering is enabled through bit 6 of the Network Configuration register
- The frame destination address matches against the multicast hash filter
- The frame destination address is not a broadcast

### 39.6.14 IEEE 1588 Support

IEEE 1588 is a standard for precision time synchronization in local area networks. It works with the exchange of special Precision Time Protocol (PTP) frames. The PTP messages can be transported over IEEE 802.3/Ethernet, over Internet Protocol Version 4 or over Internet Protocol Version 6 as described in the annex of IEEE P1588.D2.1.

The GEMAC indicates the message timestamp point (asserted on the start packet delimiter and de-asserted at end of frame) for all frames and the passage of PTP event frames (asserted when a PTP event frame is detected and de-asserted at end of frame).

IEEE 802.1AS is a subset of IEEE 1588. One difference is that IEEE 802.1AS uses the Ethernet multicast address 0180C200000E for sync frame recognition whereas IEEE 1588 does not. GEMAC is designed to recognize sync frames with both IEEE 802.1AS and IEEE 1588 addresses and so can support both 1588 and 802.1AS frame recognition simultaneously.

Synchronization between master and slave clocks is a two stage process.

First, the offset between the master and slave clocks is corrected by the master sending a sync frame to the slave with a follow up frame containing the exact time the sync frame was sent. Hardware assist modules at the master and slave side detect exactly when the sync frame was sent by the master and received by the slave. The slave then corrects its clock to match the master clock.

Second, the transmission delay between the master and slave is corrected. The slave sends a delay request frame to the master which sends a delay response frame in reply. Hardware assist modules at the master and slave side detect exactly when the delay request frame was sent by the slave and received by the master. The slave will now have enough information to adjust its clock to account for delay. For example, if the slave was assuming zero delay, the actual delay will be half the difference between the transmit and receive time of the delay request frame (assuming equal transmit and receive times) because the slave clock will be lagging the master clock by the delay time already.

The timestamp is taken when the message timestamp point passes the clock timestamp point. This can generate an interrupt if enabled (GMAC_IER). However, MAC Filtering configuration is needed to actually ‘copy’ the message to memory. For Ethernet, the message timestamp point is the SFD and the clock timestamp point is the MII interface. (The IEEE 1588 specification refers to sync and delay_req messages as event messages as these require timestamping. These events are captured in the registers

GMAC_EFTx and GMAC_EFRx, respectively. Follow up, delay response and management messages do not require timestamping and are referred to as general messages.)

1588 version 2 defines two additional PTP event messages. These are the peer delay request (Pdelay_Req) and peer delay response (Pdelay_Resp) messages. These events are captured in the registers GMAC_PEFTx and GMAC_PEFRx, respectively. These messages are used to calculate the delay on a link. Nodes at both ends of a link send both types of frames (regardless of whether they contain a master or slave clock). The Pdelay_Resp message contains the time at which a Pdelay_Req was received and is itself an event message. The time at which a Pdelay_Resp message is received is returned in a Pdelay_Resp_Follow_Up message.

1588 version 2 introduces transparent clocks of which there are two kinds, peer-to-peer (P2P) and end-to-end (E2E). Transparent clocks measure the transit time of event messages through a bridge and amend a correction field within the message to allow for the transit time. P2P transparent clocks additionally correct for the delay in the receive path of the link using the information gathered from the peer delay frames. With P2P transparent clocks delay_req messages are not used to measure link delay. This simplifies the protocol and makes larger systems more stable.

The GEMAC recognizes four different encapsulations for PTP event messages:

1. 1588 version 1 (UDP/IPv4 multicast)
2. 1588 version 2 (UDP/IPv4 multicast)
3. 1588 version 2 (UDP/IPv6 multicast)
4. 1588 version 2 (Ethernet multicast)

**Table 39-6. Example of Sync Frame in 1588 Version 1 Format**

| Frame Segment | Value |
|---|---|
| Preamble/SFD | 55555555555555D5 |
| DA (Octets 0–5) | – |
| SA (Octets 6–11) | – |
| Type (Octets 12–13) | 0800 |
| IP stuff (Octets 14–22) | – |
| UDP (Octet 23) | 11 |
| IP stuff (Octets 24–29) | – |
| IP DA (Octets 30–32) | E00001 |
| IP DA (Octet 33) | 81 or 82 or 83 or 84 |
| Source IP port (Octets 34–35) | – |
| Dest IP port (Octets 36–37) | 013F |
| Other stuff (Octets 38–42) | – |
| Version PTP (Octet 43) | 01 |
| Other stuff (Octets 44–73) | – |
| Control (Octet 74) | 00 |
| Other stuff (Octets 75–168) | – |

**Table 39-7. Example of Delay Request Frame in 1588 Version 1 Format**

| Frame Segment | Value |
|---|---|
| Preamble/SFD | 55555555555555D5 |
| DA (Octets 0–5) | – |
| SA (Octets 6–11) | – |
| Type (Octets 12–13) | 0800 |
| IP stuff (Octets 14–22) | – |
| UDP (Octet 23) | 11 |
| IP stuff (Octets 24–29) | – |
| IP DA (Octets 30–32) | E00001 |
| IP DA (Octet 33) | 81 or 82 or 83 or 84 |
| Source IP port (Octets 34–35) | – |
| Dest IP port (Octets 36–37) | 013F |
| Other stuff (Octets 38–42) | – |
| Version PTP (Octet 43) | 01 |
| Other stuff (Octets 44–73) | – |
| Control (Octet 74) | 01 |
| Other stuff (Octets 75–168) | – |

For 1588 version 1 messages, sync and delay request frames are indicated by the GEMAC if the frame type field indicates TCP/IP, UDP protocol is indicated, the destination IP address is 224.0.1.129/130/131 or 132, the destination UDP port is 319 and the control field is correct.

The control field is 0x00 for sync frames and 0x01 for delay request frames.

For 1588 version 2 messages, the type of frame is determined by looking at the message type field in the first byte of the PTP frame. Whether a frame is version 1 or version 2 can be determined by looking at the version PTP field in the second byte of both version 1 and version 2 PTP frames.

In version 2 messages sync frames have a message type value of 0x0, delay_req have 0x1, Pdelay_Req have 0x2 and Pdelay_Resp have 0x3.

**Table 39-8. Example of Sync Frame in 1588 Version 2 (UDP/IPv4) Format**

| Frame Segment | Value |
|---|---|
| Preamble/SFD | 55555555555555D5 |
| DA (Octets 0–5) | – |
| SA (Octets 6–11) | – |
| Type (Octets 12–13) | 0800 |
| IP stuff (Octets 14–22) | – |

| .....continued | |
|---|---|
| **Frame Segment** | **Value** |
| UDP (Octet 23) | 11 |
| IP stuff (Octets 24–29) | – |
| IP DA (Octets 30–33) | E0000181 |
| Source IP port (Octets 34–35) | – |
| Dest IP port (Octets 36–37) | 013F |
| Other stuff (Octets 38–41) | – |
| Message type (Octet 42) | 00 |
| Version PTP (Octet 43) | 02 |

**Table 39-9. Example of Pdelay_Req Frame in 1588 Version 2 (UDP/IPv4) Format**

| Frame Segment | Value |
|---|---|
| Preamble/SFD | 55555555555555D5 |
| DA (Octets 0–5) | – |
| SA (Octets 6–11) | – |
| Type (Octets 12–13) | 0800 |
| IP stuff (Octets 14–22) | – |
| UDP (Octet 23) | 11 |
| IP stuff (Octets 24–29) | – |
| IP DA (Octets 30–33) | E000006B |
| Source IP port (Octets 34–35) | – |
| Dest IP port (Octets 36–37) | 013F |
| Other stuff (Octets 38–41) | – |
| Message type (Octet 42) | 02 |
| Version PTP (Octet 43) | 02 |

**Table 39-10. Example of Sync Frame in 1588 Version 2 (UDP/IPv6) Format**

| Frame Segment | Value |
|---|---|
| Preamble/SFD | 55555555555555D5 |
| DA (Octets 0–5) | – |
| SA (Octets 6–11) | – |
| Type (Octets 12–13) | 86dd |
| IP stuff (Octets 14–19) | – |

| .....continued | |
|---|---|
| **Frame Segment** | **Value** |
| UDP (Octet 20) | 11 |
| IP stuff (Octets 21–37) | – |
| IP DA (Octets 38–53) | FF0X00000000018 |
| Source IP port (Octets 54–55) | – |
| Dest IP port (Octets 56–57) | 013F |
| Other stuff (Octets 58–61) | – |
| Message type (Octet 62) | 00 |
| Other stuff (Octets 63–93) | – |
| Version PTP (Octet 94) | 02 |

**Table 39-11. Example of Pdelay_Resp Frame in 1588 Version 2 (UDP/IPv6) Format**

| Frame Segment | Value |
|---|---|
| Preamble/SFD | 55555555555555D5 |
| DA (Octets 0–5) | – |
| SA (Octets 6–11) | – |
| Type (Octets 12–13) | 86dd |
| IP stuff (Octets 14–19) | – |
| UDP (Octet 20) | 11 |
| IP stuff (Octets 21–37) | – |
| IP DA (Octets 38–53) | FF0200000000006B |
| Source IP port (Octets 54–55) | – |
| Dest IP port (Octets 56–57) | 013F |
| Other stuff (Octets 58–61) | – |
| Message type (Octet 62) | 03 |
| Other stuff (Octets 63–93) | – |
| Version PTP (Octet 94) | 02 |

For the multicast address 011B19000000 sync and delay request frames are recognized depending on the message type field, 00 for sync and 01 for delay request.

**Table 39-12. Example of Sync Frame in 1588 Version 2 (Ethernet Multicast) Format**

| Frame Segment | Value |
|---|---|
| Preamble/SFD | 55555555555555D5 |
| DA (Octets 0–5) | 011B19000000 |

| .....continued | |
|---|---|
| **Frame Segment** | **Value** |
| SA (Octets 6–11) | – |
| Type (Octets 12–13) | 88F7 |
| Message type (Octet 14) | 00 |
| Version PTP (Octet 15) | 02 |

Pdelay request frames need a special multicast address so they can pass through ports blocked by the spanning tree protocol. For the multicast address 0180C200000E sync, Pdelay_Req and Pdelay_Resp frames are recognized depending on the message type field, 00 for sync, 02 for pdelay request and 03 for pdelay response.

**Table 39-13. Example of Pdelay_Req Frame in 1588 Version 2 (Ethernet Multicast) Format**

| Frame Segment | Value |
|---|---|
| Preamble/SFD | 55555555555555D5 |
| DA (Octets 0–5) | 0180C200000E |
| SA (Octets 6–11) | – |
| Type (Octets 12–13) | 88F7 |
| Message type (Octet 14) | 00 |
| Version PTP (Octet 15) | 02 |

### 39.6.15 Timestamp Unit

The TSU consists of a timer and registers to capture the time at which PTP event frames cross the message timestamp point. An interrupt is issued when a capture register is updated.

The timer is implemented as a 94-bit register with the upper 48 bits counting seconds, the next 30 bits counting nanoseconds and the lowest 16 bits counting sub-nanoseconds. The lower 46 bits rolls over when they have counted to one second. An interrupt is generated when the seconds increment. The timer value can be read, written and adjusted through the APB interface. The timer is clocked by MCK.

The amount by which the timer increments each clock cycle is controlled by the timer increment registers (GMAC_TI). Bits 7:0 are the default increment value in nanoseconds and an additional 16 bits of sub-nanosecond resolution are available using the Timer Increment Sub-nanoseconds register (GMAC_TISUBN). If the rest of the register is written with zero, the timer increments by the value in [7:0], plus the value of GMAC_TISUBN, each clock cycle.

The GMAC_TISUBN register allows a resolution of approximately 15 femtoseconds.

Bits 15:8 of the increment register are the alternative increment value in nanoseconds and bits 23:16 are the number of increments after which the alternative increment value is used. If 23:16 are zero then the alternative increment value will never be used.

Taking the example of 10.2 MHz, there are 102 cycles every ten microseconds or 51 every five microseconds. So a timer with a 10.2 MHz clock source is constructed by incrementing by 98 ns for fifty cycles and then incrementing by 100 ns (98 × 50 + 100 = 5000). This is programmed by setting the 1588 Timer Increment register to 0x00326462.

For a 49.8 MHz clock source it would be 20 ns for 248 cycles followed by an increment of 40 ns (20 × 248 + 40 = 5000) programmed as 0x00F82814.

Having eight bits for the “number of increments” field allows frequencies up to 50 MHz to be supported with 200 kHz resolution.

Without the alternative increment field the period of the clock would be limited to an integer number of nanoseconds, resulting in supported clock frequencies of 8, 10, 20, 25, 40, 50, 100, 125, 200 and 250 MHz.

There are eight additional 80-bit registers that capture the time at which PTP event frames are transmitted and received. An interrupt is issued when these registers are updated. The TSU timer count value can be compared to a programmable comparison value. For the comparison, the 48 bits of the seconds value and the upper 22 bits of the nanoseconds value are used. A signal (GTSUCOMP) is provided to indicate when the TSU timer count value is equal to the comparison value stored in the TSU timer comparison value registers (0x0DC, 0x0E0, and 0x0E4). The GTSUCOMP signal can be routed to the Timer peripheral to automatically toggle pin TOIA11/PD21. This can be used as the reference clock for an external PLL to regenerate the audio clock in Ethernet AVB. An interrupt can also be generated (if enabled) when the TSU timer count value and comparison value are equal, mapped to bit 29 of the Interrupt Status register.

### 39.6.16 MAC 802.3 Pause Frame Support

**Note:** See Clause 31, and Annex 31A and 31B of the IEEE standard 802.3 for a full description of MAC 802.3 pause operation.

The following table shows the start of a MAC 802.3 pause frame.

**Table 39-14. Start of an 802.3 Pause Frame**

| Address | | Type (MAC Control Frame) | Pause | |
|---|---|---|---|---|
| Destination | Source | | Opcode | Time |
| 0x0180C2000001 | 6 bytes | 0x8808 | 0x0001 | 2 bytes |

The GEMAC supports both hardware controlled pause of the transmitter, upon reception of a pause frame, and hardware generated pause frame transmission.

#### 39.6.16.1 802.3 Pause Frame Reception

Bit 13 of the Network Configuration register is the pause enable control for reception. If this bit is set, transmission pauses if a non zero pause quantum frame is received.

If a valid pause frame is received, then the Pause Time register is updated with the new frame's pause time, regardless of whether a previous pause frame is active or not. An interrupt (either bit 12 or bit 13 of the Interrupt Status register) is triggered when a pause frame is received, but only if the interrupt has been enabled (bit 12 and bit 13 of the Interrupt Mask register). Pause frames received with non zero quantum are indicated through the interrupt bit 12 of the Interrupt Status register. Pause frames received with zero quantum are indicated on bit 13 of the Interrupt Status register.

Once the Pause Time register is loaded and the frame currently being transmitted has been sent, no new frames are transmitted until the pause time reaches zero. The loading of a new pause time, and hence the pausing of transmission, only occurs when the GEMAC is configured for full duplex operation. If the GEMAC is configured for half duplex there will be no transmission pause, but the pause frame received interrupt will still be triggered. A valid pause frame is defined as having a destination address that matches either the address stored in Specific Address 1 register or if it matches the reserved address of

0x0180C2000001. It must also have the MAC control frame type ID of 0x8808 and have the pause opcode of 0x0001.

Pause frames that have frame check sequence (FCS) or other errors will be treated as invalid and will be discarded. 802.3 Pause frames that are received after Priority-based Flow Control (PFC) has been negotiated will also be discarded. Valid pause frames received will increment the Pause Frames Received statistic register.

The Pause Time register decrements every 512 bit times once transmission has stopped. For test purposes, the retry test bit can be set (bit 12 in the Network Configuration register) which causes the Pause Time register to decrement every GTXCK cycle once transmission has stopped.

The interrupt (bit 13 in the Interrupt Status register) is asserted whenever the Pause Time register decrements to zero (assuming it has been enabled by bit 13 in the Interrupt Mask register). This interrupt is also set when a zero quantum pause frame is received.

#### 39.6.16.2 802.3 Pause Frame Transmission

Automatic transmission of pause frames is supported through the transmit pause frame bits of the Network Control register. If either bit 11 or bit 12 of the Network Control register is written with logic 1, an 802.3 pause frame will be transmitted, providing full duplex is selected in the Network Configuration register and the transmit block is enabled in the Network Control register.

Pause frame transmission will happen immediately if transmit is inactive or if transmit is active between the current frame and the next frame due to be transmitted.

Transmitted pause frames comprise the following:

- A destination address of 01-80-C2-00-00-01
- A source address taken from Specific Address 1 register
- A type ID of 88-08 (MAC control frame)
- A pause opcode of 00-01
- A Pause Quantum register
- Fill of 00 to take the frame to minimum frame length
- Valid FCS

The pause quantum used in the generated frame will depend on the trigger source for the frame as follows:

- If bit 11 is written with a one, the pause quantum will be taken from the Transmit Pause Quantum register. The Transmit Pause Quantum register resets to a value of 0xFFFF giving maximum pause quantum as default.
- If bit 12 is written with a one, the pause quantum will be zero.

After transmission, a pause frame transmitted interrupt will be generated (bit 14 of the Interrupt Status register) and the only the statistics register Pause Frames Transmitted is incremented.

Pause frames can also be transmitted by the MAC using normal frame transmission methods.

### 39.6.17 MAC PFC Priority-based Pause Frame Support

**Note:** Refer to the 802.1Qbb standard for a full description of priority-based pause operation.

The following table shows the start of a Priority-based Flow Control (PFC) pause frame.

**Table 39-15. Start of a PFC Pause Frame**

| Address | | Type (Mac Control Frame) | Pause Opcode | Priority Enable Vector | Pause Time |
|---|---|---|---|---|---|
| Destination | Source | | | | |
| 0x0180C2000001 | 6 bytes | 0x8808 | 0x1001 | 2 bytes | 8 × 2 bytes |

The GEMAC supports PFC priority-based pause transmission and reception. Before PFC pause frames can be received, bit 16 of the Network Control register must be set.

#### 39.6.17.1 PFC Pause Frame Reception

The ability to receive and decode priority-based pause frames is enabled by setting bit 16 of the Network Control register. When this bit is set, the GEMAC will match either classic 802.3 pause frames or PFC priority-based pause frames. Once a priority-based pause frame has been received and matched, then from that moment on the GEMAC will only match on priority-based pause frames (this is an 802.1Qbb requirement, known as PFC negotiation). Once priority-based pause has been negotiated, any received 802.3x format pause frames will not be acted upon.

If a valid priority-based pause frame is received then the GEMAC will decode the frame and determine which, if any, of the eight priorities require to be paused. Up to eight Pause Time registers are then updated with the eight pause times extracted from the frame regardless of whether a previous pause operation is active or not. An interrupt (either bit 12 or bit 13 of the Interrupt Status register) is triggered when a pause frame is received, but only if the interrupt has been enabled (bit 12 and bit 13 of the Interrupt Mask register). Pause frames received with non zero quantum are indicated through the interrupt bit 12 of the Interrupt Status register. Pause frames received with zero quantum are indicated on bit 13 of the Interrupt Status register. The loading of a new pause time only occurs when the GEMAC is configured for full duplex operation. If the GEMAC is configured for half duplex, the pause time counters will not be loaded, but the pause frame received interrupt will still be triggered. A valid pause frame is defined as having a destination address that matches either the address stored in Specific Address 1 register or if it matches the reserved address of 0x0180C2000001. It must also have the MAC control frame type ID of 0x8808 and have the pause opcode of 0x0101.

Pause frames that have frame check sequence (FCS) or other errors will be treated as invalid and will be discarded. Valid pause frames received will increment the Pause Frames Received Statistic register.

The Pause Time registers decrement every 512 bit times immediately following the PFC frame reception. For test purposes, the retry test bit can be set (bit 12 in the Network Configuration register) which causes the Pause Time register to decrement every GRXCK cycle once transmission has stopped.

The interrupt (bit 13 in the Interrupt Status register) is asserted whenever the Pause Time register decrements to zero (assuming it has been enabled by bit 13 in the Interrupt Mask register). This interrupt is also set when a zero quantum pause frame is received.

#### 39.6.17.2 PFC Pause Frame Transmission

Automatic transmission of pause frames is supported through the transmit priority-based pause frame bit of the Network Control register. If bit 17 of the Network Control register is written with logic 1, a PFC pause frame will be transmitted providing full duplex is selected in the Network Configuration register and the transmit block is enabled in the Network Control register. When bit 17 of the Network Control register is set, the fields of the priority-based pause frame will be built using the values stored in the Transmit PFC Pause register.

Pause frame transmission will happen immediately if transmit is inactive or if transmit is active between the current frame and the next frame due to be transmitted.

Transmitted pause frames comprise the following:

- A destination address of 01-80-C2-00-00-01
- A source address taken from Specific Address 1 register
- A type ID of 88-08 (MAC control frame)
- A pause opcode of 01-01
- A priority enable vector taken from Transmit PFC Pause register
- 8 Pause Quantum registers
- Fill of 00 to take the frame to minimum frame length
- Valid FCS

The Pause Quantum registers used in the generated frame will depend on the trigger source for the frame as follows:

- If bit 17 of the Network Control register is written with a one, then the priority enable vector of the priority-based pause frame will be set equal to the value stored in the Transmit PFC Pause register [7:0]. For each entry equal to zero in the Transmit PFC Pause register [15:8], the pause quantum field of the pause frame associated with that entry will be taken from the Transmit Pause Quantum register. For each entry equal to one in the Transmit PFC Pause register [15:8], the pause quantum associated with that entry will be zero.
- The Transmit Pause Quantum register resets to a value of 0xFFFF giving maximum pause quantum as default.

After transmission, a pause frame transmitted interrupt will be generated (bit 14 of the Interrupt Status register) and the only statistics register that will be incremented will be the Pause Frames Transmitted register.

PFC Pause frames can also be transmitted by the MAC using normal frame transmission methods.

### 39.6.18 Energy-efficient Ethernet Support

IEEE 802.3az adds support for energy efficiency to Ethernet. These are the key features of 802.3az:

- Allows a system’s transmit path to enter a low power mode if there is nothing to transmit.
- Allows a PHY to detect whether its link partner’s transmit path is in low power mode, therefore allowing the system’s receive path to enter low power mode.
- Link remains up during lower power mode and no frames are dropped.
- Asymmetric, one direction can be in low power mode while the other is transmitting normally.
- LPI (Low Power Idle) signaling is used to control entry and exit to and from low power modes.
- LPI signaling can only take place if both sides have indicated support for it through auto-negotiation.

These are the key features of 802.3az operation:

- Low power control is done at the MII (reconciliation sublayer).
- As an architectural convenience in writing the 802.3az it is assumed that transmission is deferred by asserting carrier sense, in practice it will not be done this way. This system will know when it has nothing to transmit and only enter low power mode when it is not transmitting.
- LPI should not be requested unless the link has been up for at least one second.
- LPI is signaled on the transmit path by asserting 0x01 on txd with tx_en low and tx_er high.
- A PHY on seeing LPI requested on the MII will send the sleep signal before going quiet. After going quiet it will periodically transmit refresh signals.

- LPI mode ends by transmitting normal idle for the wake time. There is a default time for this but it can be adjusted in software using the Link Layer Discovery Protocol (LLDP) described in Clause 79 of 802.3az.
- LPI is indicated at the receive side when sleep and refresh signaling has been detected.

### 39.6.19 802.1Qav Support - Credit-based Shaping

A credit-based shaping algorithm is available on the two highest priority queues and is defined in the standard 802.1Qav: Forwarding and Queuing Enhancements for Time-Sensitive Streams. This allows traffic on these queues to be limited and to allow other queues to transmit.

Traffic shaping is enabled via the CBS (Credit Based Shaping) Control register. This enables a counter which stores the amount of transmit 'credit', measured in bytes that a particular queue has. A queue may only transmit if it has non-negative credit. If a queue has data to send, but is held off from doing as another queue is transmitting, then credit will accumulate in the credit counter at the rate defined in the IdleSlope register (GMAC_CBSISQx) for that queue.

**portTransmitRate** is the transmission rate, in bits per second, that the underlying MAC service that supports transmission through the Port provides. The value of this parameter is determined by the operation of the MAC.

**IdleSlope** is the rate of change of increasing credit when waiting to transmit and must be less than the value of the portTransmitRate.

The max value of IdleSlope (or sendSlope) is (portTransmitRate / bits_per_MII_Clock).

In case of 100Mbps, maximum IdleSlope = (100Mbps / 4) = 0x17D7840.

When this queue is transmitting, the credit counter is decremented at the rate of sendSlope, which is defined as (portTransmitRate - IdleSlope). A queue can accumulate negative credit when transmitting which will hold off any other transfers from that queue until credit returns to a non-negative value. No transfers are halted when a queue's credit becomes negative; it will accumulate negative credit until the transfer completes.

The highest priority queue always has priority regardless of which queue has the most credit.

### 39.6.20 LPI Operation in the GEMAC

It is best to use firmware to control LPI. LPI operation happens at the system level. Firmware gives maximum control and flexibility of operation. LPI operation is straightforward and firmware should be capable of responding within the required timeframes.

Autonegotiation:

1. Indicate EEE capability using next page autonegotiation.

For the transmit path:

1. If the link has been up for 1 second and there is nothing being transmitted, write to the TXLPIEN bit in the Network Control register.
2. Wake up by clearing the TXLPIEN bit in the Network Control register.

For the receive path:

1. Enable RXLPISBC bit in GMAC_IER. The bit RXLPIS is set in Network Status Register triggering an interrupt.
2. Wait for an interrupt to indicate that LPI has been received.
3. Disable relevant parts of the receive path if desired.

4. The RXLPIS bit in Network Status Register gets cleared to indicate that regular idle has been received. This triggers an interrupt.
5. Re-enable the receive path.

### 39.6.21 PHY Interface

Different PHY interfaces are supported by the Ethernet MAC:

- MII
- RMII

The MII interface is provided for 10/100 operation and uses txd[3:0] and rxd[3:0]. The RMII interface is provided for 10/100 operation and uses txd[1:0] and rxd[1:0].

### 39.6.22 10/100 Operation

The 10/100Mbps speed bit in the Network Configuration register is used to select between 10Mbps and 100Mbps.

### 39.6.23 Jumbo Frames

The jumbo frames enable bit in the Network Configuration register allows the GEMAC, in its default configuration, to receive jumbo frames up to 10240 bytes in size. This operation does not form part of the IEEE 802.3 specification and is normally disabled. When jumbo frames are enabled, frames received with a frame size greater than 10240 bytes are discarded.

## 39.7 Programming Interface

### 39.7.1 Initialization

#### 39.7.1.1 Configuration

Initialization of the GEMAC configuration (e.g., loop back mode, frequency ratios) must be done while the transmit and receive circuits are disabled. See the description of the Network Control register and Network Configuration register earlier in this document.

To change loop back mode, the following sequence of operations must be followed:

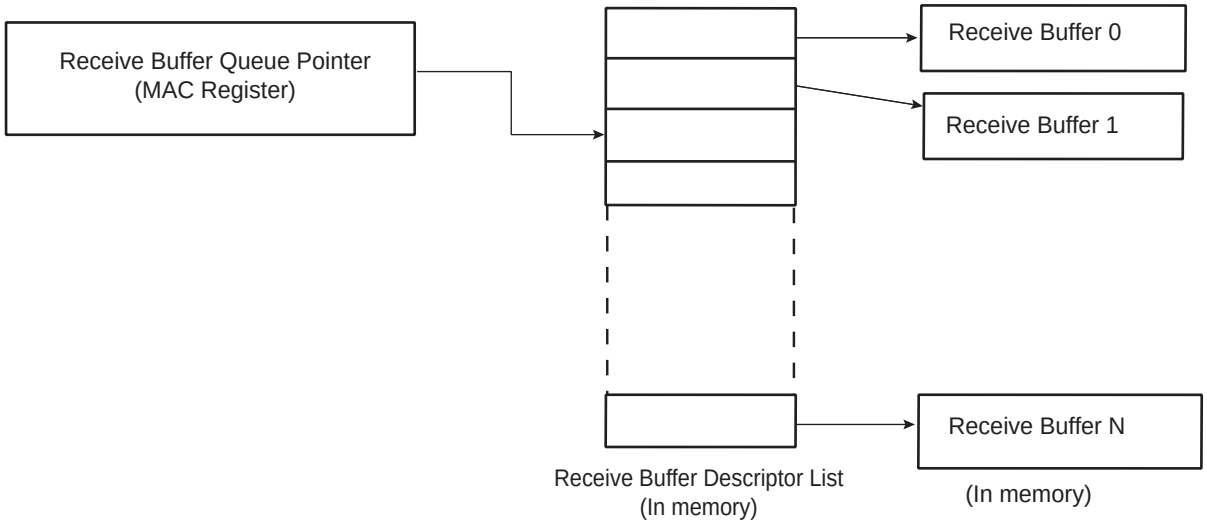
1. Write to Network Control register to disable transmit and receive circuits.
2. Write to Network Control register to change loop back mode.
3. Write to Network Control register to re-enable transmit or receive circuits.
   **Note:** These writes to the Network Control register cannot be combined in any way.

#### 39.7.1.2 Receive Buffer List

Receive data is written to areas of data (i.e., buffers) in system memory. These buffers are listed in another data structure that also resides in main memory. This data structure (receive buffer queue) is a sequence of descriptor entries as defined in Receive Buffer Descriptor Entry.

The Receive Buffer Queue Pointer register points to this data structure.

**Figure 39-3. Receive Buffer List**




To create the list of buffers:

1. Allocate a number (N) of buffers of X bytes in system memory, where X is the DMA buffer length programmed in the DMA Configuration register.
2. Allocate an area 8N bytes for the receive buffer descriptor list in system memory and create N entries in this list. Mark all entries in this list as owned by GEMAC, i.e., bit 0 of word 0 set to 0.
3. Mark the last descriptor in the queue with the wrap bit (bit 1 in word 0 set to 1).
4. Write address of receive buffer descriptor list and control information to GEMAC register receive buffer queue pointer
5. The receive circuits can then be enabled by writing to the address recognition registers and the Network Control register.
   **Note:** The queue pointers must be initialized and point to USED descriptors for all queues including those not intended for use.

#### 39.7.1.3 Transmit Buffer List

Transmit data is read from areas of data (the buffers) in system memory. These buffers are listed in another data structure that also resides in main memory. This data structure (Transmit Buffer Queue) is a sequence of descriptor entries as defined in Transmit Buffer Descriptor Entry.

The Transmit Buffer Queue Pointer register points to this data structure.

To create this list of buffers:

1. Allocate a number (N) of buffers of between 1 and 2047 bytes of data to be transmitted in system memory. Up to 128 buffers per frame are allowed.
2. Allocate an area 8N bytes for the transmit buffer descriptor list in system memory and create N entries in this list. Mark all entries in this list as owned by GEMAC, i.e., bit 31 of word 1 set to 0.
3. Mark the last descriptor in the queue with the wrap bit (bit 30 in word 1 set to 1).
4. Write address of transmit buffer descriptor list and control information to GEMAC register transmit buffer queue pointer.
5. The transmit circuits can then be enabled by writing to the Network Control register.
   **Note:** The queue pointers must be initialized and point to USED descriptors for all queues including those not intended for use.

#### 39.7.1.4 Address Matching

The GEMAC Hash register pair and the four Specific Address register pairs must be written with the required values. Each register pair comprises of a bottom register and top register, with the bottom register being written first. The address matching is disabled for a particular register pair after the bottom register has been written and re-enabled when the top register is written. Each register pair may be written at any time, regardless of whether the receive circuits are enabled or disabled.

As an example, to set Specific Address 1 register to recognize destination address 21:43:65:87:A9:CB, the following values are written to Specific Address 1 Bottom register and Specific Address 1 Top register:

- Specific Address 1 Bottom register bits 31:0 (0x98): 0x8765_4321.
- Specific Address 1 Top register bits 31:0 (0x9C): 0x0000_CBA9.

**Note:** The address matching is the first level of filtering. If there is a match, the screeners are the next level of filtering for routing the data to the appropriate queue. See Priority Queueing in the DMA for more details.

#### 39.7.1.5 PHY Maintenance

The PHY Maintenance register is implemented as a shift register. Writing to the register starts a shift operation which is signalled as complete when bit two is set in the Network Status register (about 2000 MCK cycles later when bits 18:16 are set to 010 in the Network Configuration register). An interrupt is generated as this bit is set.

During this time, the MSB of the register is output on the MDIO pin and the LSB updated from the MDIO pin with each Management Data Clock (MDC) cycle. This causes the transmission of a PHY management frame on MDIO. See section 22.2.4.5 of the IEEE 802.3 standard.

Reading during the shift operation will return the current contents of the shift register. At the end of the management operation the bits will have shifted back to their original locations. For a read operation the data bits are updated with data read from the PHY. It is important to write the correct values to the register to ensure a valid PHY management frame is produced.

The Management Data Clock (MDC) should not toggle faster than 2.5 MHz (minimum period of 400 ns), as defined by the IEEE 802.3 standard. MDC is generated by dividing down MCK. Three bits in the Network Configuration register determine by how much MCK should be divided to produce MDC.

#### 39.7.1.6 Interrupts

There are 18 interrupt conditions that are detected within the GEMAC. The conditions are ORed to make multiple interrupts. Depending on the overall system design this may be passed through a further level of interrupt collection (interrupt controller). On receipt of the interrupt signal, the CPU enters the interrupt handler. Refer to the device interrupt controller documentation to identify that it is the GEMAC that is generating the interrupt. To ascertain which interrupt, read the Interrupt Status register. Note that in the default configuration this register will clear itself after being read, though this may be configured to be write-one-to-clear if desired.

At reset all interrupts are disabled. To enable an interrupt, write to Interrupt Enable register with the pertinent interrupt bit set to 1. To disable an interrupt, write to Interrupt Disable register with the pertinent interrupt bit set to 1. To check whether an interrupt is enabled or disabled, read Interrupt Mask register. If the bit is set to 1, the interrupt is disabled.

#### 39.7.1.7 Transmitting Frames

The procedure to set up a frame for transmission is the following:

1. Enable transmit in the Network Control register.

2. Allocate an area of system memory for transmit data. This does not have to be contiguous, varying byte lengths can be used if they conclude on byte borders.
3. Set-up the transmit buffer list by writing buffer addresses to word zero of the transmit buffer descriptor entries and control and length to word one.
4. Write data for transmission into the buffers pointed to by the descriptors.
5. Write the address of the first buffer descriptor to transmit buffer descriptor queue pointer.
6. Enable appropriate interrupts.
7. Write to the transmit start bit (TSTART) in the Network Control register.

#### 39.7.1.8 Receiving Frames

When a frame is received and the receive circuits are enabled, the GEMAC checks the address and, in the following cases, the frame is written to system memory:

- If it matches one of the four Specific Address registers.
- If it matches one of the four Type ID registers.
- If it matches the hash address function.
- If it is a broadcast address (0xFFFFFFFFFFFF) and broadcasts are allowed.
- If the GEMAC is configured to “copy all frames”.

The register receive buffer queue pointer points to the next entry in the receive buffer descriptor list and the GEMAC uses this as the address in system memory to write the frame to.

Once the frame has been completely and successfully received and written to system memory, the GEMAC then updates the receive buffer descriptor entry (see Receive Buffer Descriptor Entry) with the reason for the address match and marks the area as being owned by software. Once this is complete, a receive complete interrupt is set. Software is then responsible for copying the data to the application area and releasing the buffer (by writing the ownership bit back to 0).

If the GEMAC is unable to write the data at a rate to match the incoming frame, then a receive overrun interrupt is set. If there is no receive buffer available, i.e., the next buffer is still owned by software, a receive buffer not available interrupt is set. If the frame is not successfully received, a statistics register is incremented and the frame is discarded without informing software.

### 39.7.2 Statistics Registers

Statistics registers are described in the User Interface beginning with GMAC Octets Transmitted Low Register and ending with GMAC UDP Checksum Errors Register.

The statistics register block begins at 0x100 and runs to 0x1B0, and comprises the registers listed below.

| | |
|---|---|
| Octets Transmitted Low Register | Broadcast Frames Received Register |
| Octets Transmitted High Register | Multicast Frames Received Register |
| Frames Transmitted Register | Pause Frames Received Register |
| Broadcast Frames Transmitted Register | 64 Byte Frames Received Register |
| Multicast Frames Transmitted Register | 65 to 127 Byte Frames Received Register |
| Pause Frames Transmitted Register | 128 to 255 Byte Frames Received Register |
| 64 Byte Frames Transmitted Register | 256 to 511 Byte Frames Received Register |
| 65 to 127 Byte Frames Transmitted Register | 512 to 1023 Byte Frames Received Register |

| | |
|---|---|
| 128 to 255 Byte Frames Transmitted Register | 1024 to 1518 Byte Frames Received Register |
| 256 to 511 Byte Frames Transmitted Register | 1519 to Maximum Byte Frames Received Register |
| 512 to 1023 Byte Frames Transmitted Register | Undersize Frames Received Register |
| 1024 to 1518 Byte Frames Transmitted Register | Oversize Frames Received Register |
| Greater Than 1518 Byte Frames Transmitted Register | Jabbers Received Register |
| Transmit Underruns Register | Frame Check Sequence Errors Register |
| Single Collision Frames Register | Length Field Frame Errors Register |
| Multiple Collision Frames Register | Receive Symbol Errors Register |
| Excessive Collisions Register | Alignment Errors Register |
| Late Collisions Register | Receive Resource Errors Register |
| Deferred Transmission Frames Register | Receive Overrun Register |
| Carrier Sense Errors Register | IP Header Checksum Errors Register |
| Octets Received Low Register | TCP Checksum Errors Register |
| Octets Received High Register | UDP Checksum Errors Register |
| Frames Received Register | |

These registers reset to zero on a read and stick at all ones when they count to their maximum value. They should be read frequently enough to prevent loss of data.

The receive statistics registers are only incremented when the receive enable bit (RXEN) is set in the Network Control register.

Once a statistics register has been read, it is automatically cleared. When reading the Octets Transmitted and Octets Received registers, bits 31:0 should be read prior to bits 47:32 to ensure reliable operation.

## 39.8 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | GMAC_NCR | 7:0 | WESTAT | INCSTAT | CLRSTAT | MPE | TXEN | RXEN | LBL | |
| | | 15:8 | SRTSM | | | TXZQPF | TXPF | THALT | TSTART | BP |
| | | 23:16 | | | | | TXLPIEN | FNP | TXPBPF | ENPBPR |
| | | 31:24 | | | | | | | | |
| 0x04 | GMAC_NCFGR | 7:0 | UNIHEN | MTIHEN | NBC | CAF | JFRAME | DNVLAN | FD | SPD |
| | | 15:8 | RXBUFO[1:0] | | PEN | RTY | | | | MAXFS |
| | | 23:16 | DCPF | DBW[1:0] | | CLK[2:0] | | | RFCS | LFERD |
| | | 31:24 | | IRXER | RXBP | IPGSEN | | IRXFCS | EFRHD | RXCOEN |
| 0x08 | GMAC_NSR | 7:0 | RXLPIS | | | | | IDLE | MDIO | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | GMAC_UR | 7:0 | | | | | | | | RMII |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | GMAC_DCFGR | 7:0 | ESPA | ESMA | | FBDLO[4:0] | | | | |
| | | 15:8 | | | | | TXCOEN | TXPBMS | RXBMS[1:0] | |
| | | 23:16 | DRBS[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | DDRP |
| 0x14 | GMAC_TSR | 7:0 | | | TXCOMP | TFC | TXGO | RLE | COL | UBR |
| | | 15:8 | | | | | | | | HRESP |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | GMAC_RBQB | 7:0 | ADDR[5:0] | | | | | | | |
| | | 15:8 | ADDR[13:6] | | | | | | | |
| | | 23:16 | ADDR[21:14] | | | | | | | |
| | | 31:24 | ADDR[29:22] | | | | | | | |
| 0x1C | GMAC_TBQB | 7:0 | ADDR[5:0] | | | | | | | |
| | | 15:8 | ADDR[13:6] | | | | | | | |
| | | 23:16 | ADDR[21:14] | | | | | | | |
| | | 31:24 | ADDR[29:22] | | | | | | | |
| 0x20 | GMAC_RSR | 7:0 | | | | | HNO | RXOVR | REC | BNA |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 | GMAC_ISR | 7:0 | TCOMP | TFC | RLEX | TUR | TXUBR | RXUBR | RCOMP | MFS |
| | | 15:8 | | PFTR | PTZ | PFNZ | HRESP | ROVR | | |
| | | 23:16 | PDRSFR | PDRQFR | SFT | DRQFT | SFR | DRQFR | | |
| | | 31:24 | | | TSUTIMCOMP | WOL | RXLPISBC | SRI | PDRSFT | PDRQFT |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x28 | GMAC_IER | 7:0 | TCOMP | TFC | RLEX | TUR | TXUBR | RXUBR | RCOMP | MFS |
| | | 15:8 | EXINT | PFTR | PTZ | PFNZ | HRESP | ROVR | | |
| | | 23:16 | PDRSFR | PDRQFR | SFT | DRQFT | SFR | DRQFR | | |
| | | 31:24 | | | TSUTIMCOMP | WOL | RXLPISBC | SRI | PDRSFT | PDRQFT |
| 0x2C | GMAC_IDR | 7:0 | TCOMP | TFC | RLEX | TUR | TXUBR | RXUBR | RCOMP | MFS |
| | | 15:8 | EXINT | PFTR | PTZ | PFNZ | HRESP | ROVR | | |
| | | 23:16 | PDRSFR | PDRQFR | SFT | DRQFT | SFR | DRQFR | | |
| | | 31:24 | | | TSUTIMCOMP | WOL | RXLPISBC | SRI | PDRSFT | PDRQFT |
| 0x30 | GMAC_IMR | 7:0 | TCOMP | TFC | RLEX | TUR | TXUBR | RXUBR | RCOMP | MFS |
| | | 15:8 | EXINT | PFTR | PTZ | PFNZ | HRESP | ROVR | | |
| | | 23:16 | PDRSFR | PDRQFR | SFT | DRQFT | SFR | DRQFR | | |
| | | 31:24 | | | TSUTIMCOMP | WOL | RXLPISBC | SRI | PDRSFT | PDRQFT |
| 0x34 | GMAC_MAN | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | PHYA[0:0] | REGA[4:0] | | | | | WTN[1:0] | |
| | | 31:24 | WZO | CLTTO | OP[1:0] | | PHYA[4:1] | | | |
| 0x38 | GMAC_RPQ | 7:0 | RPQ[7:0] | | | | | | | |
| | | 15:8 | RPQ[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x3C | GMAC_TPQ | 7:0 | TPQ[7:0] | | | | | | | |
| | | 15:8 | TPQ[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x40 | GMAC_TPSF | 7:0 | TPB1ADR[7:0] | | | | | | | |
| | | 15:8 | | | | | TPB1ADR[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | ENTXP | | | | | | | |
| 0x44 | GMAC_RPSF | 7:0 | RPB1ADR[7:0] | | | | | | | |
| | | 15:8 | | | | | RPB1ADR[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | ENRXP | | | | | | | |
| 0x48 | GMAC_RJFML | 7:0 | FML[7:0] | | | | | | | |
| | | 15:8 | | | FML[13:8] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x4C ... 0x7F | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x80 | GMAC_HRB | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0x84 | GMAC_HRT | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0x88 | GMAC_SAB1 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0x8C | GMAC_SAT1 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x90 | GMAC_SAB2 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0x94 | GMAC_SAT2 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x98 | GMAC_SAB3 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0x9C | GMAC_SAT3 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xA0 | GMAC_SAB4 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0xA4 | GMAC_SAT4 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xA8 | GMAC_TIDM1 | 7:0 | TID[7:0] | | | | | | | |
| | | 15:8 | TID[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | ENID1 | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xAC | GMAC_TIDM2 | 7:0 | TID[7:0] | | | | | | | |
| | | 15:8 | TID[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | ENID2 | | | | | | | |
| 0xB0 | GMAC_TIDM3 | 7:0 | TID[7:0] | | | | | | | |
| | | 15:8 | TID[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | ENID3 | | | | | | | |
| 0xB4 | GMAC_TIDM4 | 7:0 | TID[7:0] | | | | | | | |
| | | 15:8 | TID[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | ENID4 | | | | | | | |
| 0xB8 | GMAC_WOL | 7:0 | IP[7:0] | | | | | | | |
| | | 15:8 | IP[15:8] | | | | | | | |
| | | 23:16 | | | | | MTI | SA1 | ARP | MAG |
| | | 31:24 | | | | | | | | |
| 0xBC | GMAC_IPGS | 7:0 | FL[7:0] | | | | | | | |
| | | 15:8 | FL[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xC0 | GMAC_SVLAN | 7:0 | VLAN_TYPE[7:0] | | | | | | | |
| | | 15:8 | VLAN_TYPE[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | ESVLAN | | | | | | | |
| 0xC4 | GMAC_TPFCP | 7:0 | PEV[7:0] | | | | | | | |
| | | 15:8 | PQ[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xC8 | GMAC_SAMB1 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0xCC | GMAC_SAMT1 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xD0 ... 0xDB | Reserved | | | | | | | | | |
| 0xDC | GMAC_NSC | 7:0 | NANOSEC[7:0] | | | | | | | |
| | | 15:8 | NANOSEC[15:8] | | | | | | | |
| | | 23:16 | | | NANOSEC[21:16] | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xE0 | GMAC_SCL | 7:0 | SEC[7:0] | | | | | | | |
| | | 15:8 | SEC[15:8] | | | | | | | |
| | | 23:16 | SEC[23:16] | | | | | | | |
| | | 31:24 | SEC[31:24] | | | | | | | |
| 0xE4 | GMAC_SCH | 7:0 | SEC[7:0] | | | | | | | |
| | | 15:8 | SEC[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xE8 ... 0xEB | Reserved | | | | | | | | | |
| 0xEC | GMAC_EFRSH | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xF0 | GMAC_PEFTSH | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xF4 | GMAC_PEFRSH | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xF8 ... 0xFB | Reserved | | | | | | | | | |
| 0xFC | GMAC_EFTSH | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0100 | GMAC_OTLO | 7:0 | TXO[7:0] | | | | | | | |
| | | 15:8 | TXO[15:8] | | | | | | | |
| | | 23:16 | TXO[23:16] | | | | | | | |
| | | 31:24 | TXO[31:24] | | | | | | | |
| 0x0104 | GMAC_OTHI | 7:0 | TXO[7:0] | | | | | | | |
| | | 15:8 | TXO[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0108 | GMAC_FT | 7:0 | FTX[7:0] | | | | | | | |
| | | 15:8 | FTX[15:8] | | | | | | | |
| | | 23:16 | FTX[23:16] | | | | | | | |
| | | 31:24 | FTX[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x010C | GMAC_BCFT | 7:0 | BFTX[7:0] | | | | | | | |
| | | 15:8 | BFTX[15:8] | | | | | | | |
| | | 23:16 | BFTX[23:16] | | | | | | | |
| | | 31:24 | BFTX[31:24] | | | | | | | |
| 0x0110 | GMAC_MFT | 7:0 | MFTX[7:0] | | | | | | | |
| | | 15:8 | MFTX[15:8] | | | | | | | |
| | | 23:16 | MFTX[23:16] | | | | | | | |
| | | 31:24 | MFTX[31:24] | | | | | | | |
| 0x0114 | GMAC_PFT | 7:0 | PFTX[7:0] | | | | | | | |
| | | 15:8 | PFTX[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0118 | GMAC_BFT64 | 7:0 | NFTX[7:0] | | | | | | | |
| | | 15:8 | NFTX[15:8] | | | | | | | |
| | | 23:16 | NFTX[23:16] | | | | | | | |
| | | 31:24 | NFTX[31:24] | | | | | | | |
| 0x011C | GMAC_TBFT127 | 7:0 | NFTX[7:0] | | | | | | | |
| | | 15:8 | NFTX[15:8] | | | | | | | |
| | | 23:16 | NFTX[23:16] | | | | | | | |
| | | 31:24 | NFTX[31:24] | | | | | | | |
| 0x0120 | GMAC_TBFT255 | 7:0 | NFTX[7:0] | | | | | | | |
| | | 15:8 | NFTX[15:8] | | | | | | | |
| | | 23:16 | NFTX[23:16] | | | | | | | |
| | | 31:24 | NFTX[31:24] | | | | | | | |
| 0x0124 | GMAC_TBFT511 | 7:0 | NFTX[7:0] | | | | | | | |
| | | 15:8 | NFTX[15:8] | | | | | | | |
| | | 23:16 | NFTX[23:16] | | | | | | | |
| | | 31:24 | NFTX[31:24] | | | | | | | |
| 0x0128 | GMAC_TBFT1023 | 7:0 | NFTX[7:0] | | | | | | | |
| | | 15:8 | NFTX[15:8] | | | | | | | |
| | | 23:16 | NFTX[23:16] | | | | | | | |
| | | 31:24 | NFTX[31:24] | | | | | | | |
| 0x012C | GMAC_TBFT1518 | 7:0 | NFTX[7:0] | | | | | | | |
| | | 15:8 | NFTX[15:8] | | | | | | | |
| | | 23:16 | NFTX[23:16] | | | | | | | |
| | | 31:24 | NFTX[31:24] | | | | | | | |
| 0x0130 | GMAC_GTBFT1518 | 7:0 | NFTX[7:0] | | | | | | | |
| | | 15:8 | NFTX[15:8] | | | | | | | |
| | | 23:16 | NFTX[23:16] | | | | | | | |
| | | 31:24 | NFTX[31:24] | | | | | | | |
| 0x0134 | GMAC_TUR | 7:0 | TXUNR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | TXUNR[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0138 | GMAC_SCF | 7:0 | SCOL[7:0] | | | | | | | |
| | | 15:8 | SCOL[15:8] | | | | | | | |
| | | 23:16 | | | | | | | SCOL[17:16] | |
| | | 31:24 | | | | | | | | |
| 0x013C | GMAC_MCF | 7:0 | MCOL[7:0] | | | | | | | |
| | | 15:8 | MCOL[15:8] | | | | | | | |
| | | 23:16 | | | | | | | MCOL[17:16] | |
| | | 31:24 | | | | | | | | |
| 0x0140 | GMAC_EC | 7:0 | XCOL[7:0] | | | | | | | |
| | | 15:8 | | | | | | | XCOL[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0144 | GMAC_LC | 7:0 | LCOL[7:0] | | | | | | | |
| | | 15:8 | | | | | | | LCOL[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0148 | GMAC_DTF | 7:0 | DEFT[7:0] | | | | | | | |
| | | 15:8 | DEFT[15:8] | | | | | | | |
| | | 23:16 | | | | | | | DEFT[17:16] | |
| | | 31:24 | | | | | | | | |
| 0x014C | GMAC_CSE | 7:0 | CSR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | CSR[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0150 | GMAC_ORLO | 7:0 | RXO[7:0] | | | | | | | |
| | | 15:8 | RXO[15:8] | | | | | | | |
| | | 23:16 | RXO[23:16] | | | | | | | |
| | | 31:24 | RXO[31:24] | | | | | | | |
| 0x0154 | GMAC_ORHI | 7:0 | RXO[7:0] | | | | | | | |
| | | 15:8 | RXO[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0158 | GMAC_FR | 7:0 | FRX[7:0] | | | | | | | |
| | | 15:8 | FRX[15:8] | | | | | | | |
| | | 23:16 | FRX[23:16] | | | | | | | |
| | | 31:24 | FRX[31:24] | | | | | | | |
| 0x015C | GMAC_BCFR | 7:0 | BFRX[7:0] | | | | | | | |
| | | 15:8 | BFRX[15:8] | | | | | | | |
| | | 23:16 | BFRX[23:16] | | | | | | | |
| | | 31:24 | BFRX[31:24] | | | | | | | |
| 0x0160 | GMAC_MFR | 7:0 | MFRX[7:0] | | | | | | | |
| | | 15:8 | MFRX[15:8] | | | | | | | |
| | | 23:16 | MFRX[23:16] | | | | | | | |
| | | 31:24 | MFRX[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0164 | GMAC_PFR | 7:0 | PFRX[7:0] | | | | | | | |
| | | 15:8 | PFRX[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0168 | GMAC_BFR64 | 7:0 | NFRX[7:0] | | | | | | | |
| | | 15:8 | NFRX[15:8] | | | | | | | |
| | | 23:16 | NFRX[23:16] | | | | | | | |
| | | 31:24 | NFRX[31:24] | | | | | | | |
| 0x016C | GMAC_TBFR127 | 7:0 | NFRX[7:0] | | | | | | | |
| | | 15:8 | NFRX[15:8] | | | | | | | |
| | | 23:16 | NFRX[23:16] | | | | | | | |
| | | 31:24 | NFRX[31:24] | | | | | | | |
| 0x0170 | GMAC_TBFR255 | 7:0 | NFRX[7:0] | | | | | | | |
| | | 15:8 | NFRX[15:8] | | | | | | | |
| | | 23:16 | NFRX[23:16] | | | | | | | |
| | | 31:24 | NFRX[31:24] | | | | | | | |
| 0x0174 | GMAC_TBFR511 | 7:0 | NFRX[7:0] | | | | | | | |
| | | 15:8 | NFRX[15:8] | | | | | | | |
| | | 23:16 | NFRX[23:16] | | | | | | | |
| | | 31:24 | NFRX[31:24] | | | | | | | |
| 0x0178 | GMAC_TBFR1023 | 7:0 | NFRX[7:0] | | | | | | | |
| | | 15:8 | NFRX[15:8] | | | | | | | |
| | | 23:16 | NFRX[23:16] | | | | | | | |
| | | 31:24 | NFRX[31:24] | | | | | | | |
| 0x017C | GMAC_TBFR1518 | 7:0 | NFRX[7:0] | | | | | | | |
| | | 15:8 | NFRX[15:8] | | | | | | | |
| | | 23:16 | NFRX[23:16] | | | | | | | |
| | | 31:24 | NFRX[31:24] | | | | | | | |
| 0x0180 | GMAC_TMXBFR | 7:0 | NFRX[7:0] | | | | | | | |
| | | 15:8 | NFRX[15:8] | | | | | | | |
| | | 23:16 | NFRX[23:16] | | | | | | | |
| | | 31:24 | NFRX[31:24] | | | | | | | |
| 0x0184 | GMAC_UFR | 7:0 | UFRX[7:0] | | | | | | | |
| | | 15:8 | | | | | | UFRX[9:8] | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0188 | GMAC_OFR | 7:0 | OFRX[7:0] | | | | | | | |
| | | 15:8 | | | | | | OFRX[9:8] | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x018C | GMAC_JR | 7:0 | JRX[7:0] | | | | | | | |
| | | 15:8 | | | | | | JRX[9:8] | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0190 | GMAC_FCSE | 7:0 | FCKR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | FCKR[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0194 | GMAC_LFFE | 7:0 | LFER[7:0] | | | | | | | |
| | | 15:8 | | | | | | | LFER[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0198 | GMAC_RSE | 7:0 | RXSE[7:0] | | | | | | | |
| | | 15:8 | | | | | | | RXSE[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x019C | GMAC_AE | 7:0 | AER[7:0] | | | | | | | |
| | | 15:8 | | | | | | | AER[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01A0 | GMAC_RRE | 7:0 | RXRER[7:0] | | | | | | | |
| | | 15:8 | RXRER[15:8] | | | | | | | |
| | | 23:16 | | | | | | | RXRER[17:16] | |
| | | 31:24 | | | | | | | | |
| 0x01A4 | GMAC_ROE | 7:0 | RXOVR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | RXOVR[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01A8 | GMAC_IHCE | 7:0 | HCKER[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01AC | GMAC_TCE | 7:0 | TCKER[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01B0 | GMAC_UCE | 7:0 | UCKER[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01B4 ... 0x01BB | Reserved | | | | | | | | | |
| 0x01BC | GMAC_TISUBN | 7:0 | LSBTIR[7:0] | | | | | | | |
| | | 15:8 | LSBTIR[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x01C0 | GMAC_TSH | 7:0 | TCS[7:0] | | | | | | | |
| | | 15:8 | TCS[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01C4 ... 0x01CF | Reserved | | | | | | | | | |
| 0x01D0 | GMAC_TSL | 7:0 | TCS[7:0] | | | | | | | |
| | | 15:8 | TCS[15:8] | | | | | | | |
| | | 23:16 | TCS[23:16] | | | | | | | |
| | | 31:24 | TCS[31:24] | | | | | | | |
| 0x01D4 | GMAC_TN | 7:0 | TNS[7:0] | | | | | | | |
| | | 15:8 | TNS[15:8] | | | | | | | |
| | | 23:16 | TNS[23:16] | | | | | | | |
| | | 31:24 | | | TNS[29:24] | | | | | |
| 0x01D8 | GMAC_TA | 7:0 | ITDT[7:0] | | | | | | | |
| | | 15:8 | ITDT[15:8] | | | | | | | |
| | | 23:16 | ITDT[23:16] | | | | | | | |
| | | 31:24 | ADJ | | ITDT[29:24] | | | | | |
| 0x01DC | GMAC_TI | 7:0 | CNS[7:0] | | | | | | | |
| | | 15:8 | ACNS[7:0] | | | | | | | |
| | | 23:16 | NIT[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01E0 | GMAC_EFTSL | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | RUD[23:16] | | | | | | | |
| | | 31:24 | RUD[31:24] | | | | | | | |
| 0x01E4 | GMAC_EFTN | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | RUD[23:16] | | | | | | | |
| | | 31:24 | | | RUD[29:24] | | | | | |
| 0x01E8 | GMAC_EFRSL | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | RUD[23:16] | | | | | | | |
| | | 31:24 | RUD[31:24] | | | | | | | |
| 0x01EC | GMAC_EFRN | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | RUD[23:16] | | | | | | | |
| | | 31:24 | | | RUD[29:24] | | | | | |
| 0x01F0 | GMAC_PEFTSL | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | RUD[23:16] | | | | | | | |
| | | 31:24 | RUD[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x01F4 | GMAC_PEFTN | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | RUD[23:16] | | | | | | | |
| | | 31:24 | | | RUD[29:24] | | | | | |
| 0x01F8 | GMAC_PEFRSL | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | RUD[23:16] | | | | | | | |
| | | 31:24 | RUD[31:24] | | | | | | | |
| 0x01FC | GMAC_PEFRN | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | RUD[23:16] | | | | | | | |
| | | 31:24 | | | RUD[29:24] | | | | | |
| 0x0200 ... 0x026F | Reserved | | | | | | | | | |
| 0x0270 | GMAC_RXLPI | 7:0 | COUNT[7:0] | | | | | | | |
| | | 15:8 | COUNT[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0274 | GMAC_RXLPITIME | 7:0 | LPITIME[7:0] | | | | | | | |
| | | 15:8 | LPITIME[15:8] | | | | | | | |
| | | 23:16 | LPITIME[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0278 | GMAC_TXLPI | 7:0 | COUNT[7:0] | | | | | | | |
| | | 15:8 | COUNT[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x027C | GMAC_TXLPITIME | 7:0 | LPITIME[7:0] | | | | | | | |
| | | 15:8 | LPITIME[15:8] | | | | | | | |
| | | 23:16 | LPITIME[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0280 ... 0x03FB | Reserved | | | | | | | | | |
| 0x03FC | GMAC_ISRPQ1 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0400 | GMAC_ISRPQ2 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0404 ... 0x043B | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x043C | GMAC_TBQBAPQ1 | 7:0 | TXBQBA[5:0] | | | | | | | |
| | | 15:8 | TXBQBA[13:6] | | | | | | | |
| | | 23:16 | TXBQBA[21:14] | | | | | | | |
| | | 31:24 | TXBQBA[29:22] | | | | | | | |
| 0x0440 | GMAC_TBQBAPQ2 | 7:0 | TXBQBA[5:0] | | | | | | | |
| | | 15:8 | TXBQBA[13:6] | | | | | | | |
| | | 23:16 | TXBQBA[21:14] | | | | | | | |
| | | 31:24 | TXBQBA[29:22] | | | | | | | |
| 0x0444 ... 0x047B | Reserved | | | | | | | | | |
| 0x047C | GMAC_RBQBAPQ1 | 7:0 | RXBQBA[5:0] | | | | | | | |
| | | 15:8 | RXBQBA[13:6] | | | | | | | |
| | | 23:16 | RXBQBA[21:14] | | | | | | | |
| | | 31:24 | RXBQBA[29:22] | | | | | | | |
| 0x0480 | GMAC_RBQBAPQ2 | 7:0 | RXBQBA[5:0] | | | | | | | |
| | | 15:8 | RXBQBA[13:6] | | | | | | | |
| | | 23:16 | RXBQBA[21:14] | | | | | | | |
| | | 31:24 | RXBQBA[29:22] | | | | | | | |
| 0x0484 ... 0x049B | Reserved | | | | | | | | | |
| 0x049C | GMAC_RBSRPQ1 | 7:0 | RBS[7:0] | | | | | | | |
| | | 15:8 | RBS[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04A0 | GMAC_RBSRPQ2 | 7:0 | RBS[7:0] | | | | | | | |
| | | 15:8 | RBS[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04A4 ... 0x04BB | Reserved | | | | | | | | | |
| 0x04BC | GMAC_CBSCR | 7:0 | | | | | | | QAE | QBE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04C0 | GMAC_CBSISQA | 7:0 | IS[7:0] | | | | | | | |
| | | 15:8 | IS[15:8] | | | | | | | |
| | | 23:16 | IS[23:16] | | | | | | | |
| | | 31:24 | IS[31:24] | | | | | | | |
| 0x04C4 | GMAC_CBSISQB | 7:0 | IS[7:0] | | | | | | | |
| | | 15:8 | IS[15:8] | | | | | | | |
| | | 23:16 | IS[23:16] | | | | | | | |
| | | 31:24 | IS[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x04C8 ... 0x04FF | Reserved | | | | | | | | | |
| 0x0500 | GMAC_ST1RPQ0 | 7:0 | DSTCM[3:0] | | | | | QNB[2:0] | | |
| | | 15:8 | UDPM[3:0] | | | | DSTCM[7:4] | | | |
| | | 23:16 | UDPM[11:4] | | | | | | | |
| | | 31:24 | | | UDPE | DSTCE | UDPM[15:12] | | | |
| 0x0504 | GMAC_ST1RPQ1 | 7:0 | DSTCM[3:0] | | | | | QNB[2:0] | | |
| | | 15:8 | UDPM[3:0] | | | | DSTCM[7:4] | | | |
| | | 23:16 | UDPM[11:4] | | | | | | | |
| | | 31:24 | | | UDPE | DSTCE | UDPM[15:12] | | | |
| 0x0508 | GMAC_ST1RPQ2 | 7:0 | DSTCM[3:0] | | | | | QNB[2:0] | | |
| | | 15:8 | UDPM[3:0] | | | | DSTCM[7:4] | | | |
| | | 23:16 | UDPM[11:4] | | | | | | | |
| | | 31:24 | | | UDPE | DSTCE | UDPM[15:12] | | | |
| 0x050C | GMAC_ST1RPQ3 | 7:0 | DSTCM[3:0] | | | | | QNB[2:0] | | |
| | | 15:8 | UDPM[3:0] | | | | DSTCM[7:4] | | | |
| | | 23:16 | UDPM[11:4] | | | | | | | |
| | | 31:24 | | | UDPE | DSTCE | UDPM[15:12] | | | |
| 0x0510 ... 0x053F | Reserved | | | | | | | | | |
| 0x0540 | GMAC_ST2RPQ0 | 7:0 | | VLANP[2:0] | | | | QNB[2:0] | | |
| | | 15:8 | COMPA[2:0] | | | ETHE | I2ETH[2:0] | | | VLANE |
| | | 23:16 | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| | | 31:24 | | COMPCE | COMPC[4:0] | | | | | COMPBE |
| 0x0544 | GMAC_ST2RPQ1 | 7:0 | | VLANP[2:0] | | | | QNB[2:0] | | |
| | | 15:8 | COMPA[2:0] | | | ETHE | I2ETH[2:0] | | | VLANE |
| | | 23:16 | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| | | 31:24 | | COMPCE | COMPC[4:0] | | | | | COMPBE |
| 0x0548 | GMAC_ST2RPQ2 | 7:0 | | VLANP[2:0] | | | | QNB[2:0] | | |
| | | 15:8 | COMPA[2:0] | | | ETHE | I2ETH[2:0] | | | VLANE |
| | | 23:16 | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| | | 31:24 | | COMPCE | COMPC[4:0] | | | | | COMPBE |
| 0x054C | GMAC_ST2RPQ3 | 7:0 | | VLANP[2:0] | | | | QNB[2:0] | | |
| | | 15:8 | COMPA[2:0] | | | ETHE | I2ETH[2:0] | | | VLANE |
| | | 23:16 | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| | | 31:24 | | COMPCE | COMPC[4:0] | | | | | COMPBE |
| 0x0550 | GMAC_ST2RPQ4 | 7:0 | | VLANP[2:0] | | | | QNB[2:0] | | |
| | | 15:8 | COMPA[2:0] | | | ETHE | I2ETH[2:0] | | | VLANE |
| | | 23:16 | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| | | 31:24 | | COMPCE | COMPC[4:0] | | | | | COMPBE |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0554 | GMAC_ST2RPQ5 | 7:0 | | VLANP[2:0] | | | | QNB[2:0] | | |
| | | 15:8 | COMPA[2:0] | | | ETHE | I2ETH[2:0] | | | VLANE |
| | | 23:16 | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| | | 31:24 | | COMPCE | COMPC[4:0] | | | | | COMPBE |
| 0x0558 | GMAC_ST2RPQ6 | 7:0 | | VLANP[2:0] | | | | QNB[2:0] | | |
| | | 15:8 | COMPA[2:0] | | | ETHE | I2ETH[2:0] | | | VLANE |
| | | 23:16 | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| | | 31:24 | | COMPCE | COMPC[4:0] | | | | | COMPBE |
| 0x055C | GMAC_ST2RPQ7 | 7:0 | | VLANP[2:0] | | | | QNB[2:0] | | |
| | | 15:8 | COMPA[2:0] | | | ETHE | I2ETH[2:0] | | | VLANE |
| | | 23:16 | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| | | 31:24 | | COMPCE | COMPC[4:0] | | | | | COMPBE |
| 0x0560 ... 0x05FB | Reserved | | | | | | | | | |
| 0x05FC | GMAC_IERPQ1 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0600 | GMAC_IERPQ2 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0604 ... 0x061B | Reserved | | | | | | | | | |
| 0x061C | GMAC_IDRPQ1 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0620 | GMAC_IDRPQ2 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0624 ... 0x063B | Reserved | | | | | | | | | |
| 0x063C | GMAC_IMRPQ1 | 7:0 | TCOMP | AHB | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0640 | GMAC_IMRPQ2 | 7:0 | TCOMP | AHB | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0644 ... 0x06DF | Reserved | | | | | | | | | |
| 0x06E0 | GMAC_ST2ER0 | 7:0 | COMPVAL[7:0] | | | | | | | |
| | | 15:8 | COMPVAL[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x06E4 | GMAC_ST2ER1 | 7:0 | COMPVAL[7:0] | | | | | | | |
| | | 15:8 | COMPVAL[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x06E8 | GMAC_ST2ER2 | 7:0 | COMPVAL[7:0] | | | | | | | |
| | | 15:8 | COMPVAL[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x06EC | GMAC_ST2ER3 | 7:0 | COMPVAL[7:0] | | | | | | | |
| | | 15:8 | COMPVAL[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x06F0 ... 0x06FF | Reserved | | | | | | | | | |
| 0x0700 | GMAC_ST2CW00 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x0704 | GMAC_ST2CW10 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0708 | GMAC_ST2CW01 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x070C | GMAC_ST2CW11 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0710 | GMAC_ST2CW02 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x0714 | GMAC_ST2CW12 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0718 | GMAC_ST2CW03 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x071C | GMAC_ST2CW13 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0720 | GMAC_ST2CW04 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x0724 | GMAC_ST2CW14 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0728 | GMAC_ST2CW05 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x072C | GMAC_ST2CW15 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0730 | GMAC_ST2CW06 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0734 | GMAC_ST2CW16 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0738 | GMAC_ST2CW07 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x073C | GMAC_ST2CW17 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0740 | GMAC_ST2CW08 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x0744 | GMAC_ST2CW18 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0748 | GMAC_ST2CW09 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x074C | GMAC_ST2CW19 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0750 | GMAC_ST2CW010 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0754 | GMAC_ST2CW110 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0758 | GMAC_ST2CW011 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x075C | GMAC_ST2CW111 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0760 | GMAC_ST2CW012 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x0764 | GMAC_ST2CW112 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0768 | GMAC_ST2CW013 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x076C | GMAC_ST2CW113 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0770 | GMAC_ST2CW014 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0774 | GMAC_ST2CW114 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0778 | GMAC_ST2CW015 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x077C | GMAC_ST2CW115 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0780 | GMAC_ST2CW016 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x0784 | GMAC_ST2CW116 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0788 | GMAC_ST2CW017 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x078C | GMAC_ST2CW117 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0790 | GMAC_ST2CW018 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0794 | GMAC_ST2CW118 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0798 | GMAC_ST2CW019 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x079C | GMAC_ST2CW119 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x07A0 | GMAC_ST2CW020 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x07A4 | GMAC_ST2CW120 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x07A8 | GMAC_ST2CW021 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x07AC | GMAC_ST2CW121 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x07B0 | GMAC_ST2CW022 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x07B4 | GMAC_ST2CW122 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x07B8 | GMAC_ST2CW023 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x07BC | GMAC_ST2CW123 | 7:0 | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1:1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 39.8.1 GMAC Network Control Register

**Name:** GMAC_NCR
**Offset:** 0x000
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TXLPIEN | FNP | TXPBPF | ENPBPR |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SRTSM | | | TXZQPF | TXPF | THALT | TSTART | BP |
| Access | R/W | | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WESTAT | INCSTAT | CLRSTAT | MPE | TXEN | RXEN | LBL | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

**Bit 19 – TXLPIEN** Enable LPI Transmission
When set, LPI (low power idle) is immediately transmitted.

**Bit 18 – FNP** Flush Next Packet
Flush the next packet from the external RX DPRAM. Writing one to this bit will only have an effect if the DMA is not currently writing a packet already stored in the DPRAM to memory.

**Bit 17 – TXPBPF** Transmit PFC Priority-based Pause Frame
Takes the values stored in the Transmit PFC Pause Register.

**Bit 16 – ENPBPR** Enable PFC Priority-based Pause Reception
Enables PFC Priority Based Pause Reception capabilities. Setting this bit enables PFC negotiation and recognition of priority-based pause frames.

**Bit 15 – SRTSM** Store Receive Timestamp to Memory

| Value | Description |
|---|---|
| 0 | Normal operation. |
| 1 | Causes the CRC of every received frame to be replaced with the value of the nanoseconds field of the 1588 timer that was captured as the receive frame passed the message timestamp point. Note that bit RFCS in register GMAC_NCFGR may not be set to 1 when the timer should be captured. |

**Bit 12 – TXZQPF** Transmit Zero Quantum Pause Frame
Writing one to this bit causes a pause frame with zero quantum to be transmitted.

**Bit 11 – TXPF** Transmit Pause Frame
Writing one to this bit causes a pause frame to be transmitted.

**Bit 10 – THALT** Transmit Halt
Writing one to this bit halts transmission as soon as any ongoing frame transmission ends.

**Bit 9 – TSTART** Start Transmission
Writing one to this bit starts transmission.

**Bit 8 – BP** Back pressure
If set in 10M or 100M half duplex mode, forces collisions on all received frames.

**Bit 7 – WESTAT** Write Enable for Statistics Registers
Setting this bit to one makes the statistics registers writable for functional test purposes.

**Bit 6 – INCSTAT** Increment Statistics Registers
This bit is write-only. Writing a one increments all the statistics registers by one for test purposes.

**Bit 5 – CLRSTAT** Clear Statistics Registers
This bit is write-only. Writing a one clears the statistics registers.

**Bit 4 – MPE** Management Port Enable
Set to one to enable the management port. When zero, forces MDIO to high impedance state and MDC low.

**Bit 3 – TXEN** Transmit Enable
When set, TXEN enables the GMAC transmitter to send data. When reset transmission will stop immediately, the transmit pipeline and control registers will be cleared and the Transmit Queue Pointer Register will reset to point to the start of the transmit descriptor list.

**Bit 2 – RXEN** Receive Enable
When set, RXEN enables the GMAC to receive data. When reset frame reception stops immediately and the receive pipeline will be cleared. The Receive Queue Pointer Register is unaffected.

**Bit 1 – LBL** Loop Back Local
Connects GTX to GRX, GTXEN to GRXDV and forces full duplex mode. GRXCK and GTXCK may malfunction as the GMAC is switched into and out of internal loop back. It is important that receive and transmit circuits have already been disabled when making the switch into and out of internal loop back.

© 2018 Microchip Technology Inc.

### 39.8.2 GMAC Network Configuration Register

**Name:** GMAC_NCFGR
**Offset:** 0x004
**Reset:** 0x00080000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | IRXER | RXBP | IPGSEN | | IRXFCS | EFRHD | RXCOEN |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DCPF | DBW[1:0] | | CLK[2:0] | | | RFCS | LFERD |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RXBUFO[1:0] | | PEN | RTY | | | | MAXFS |
| Access | R/W | R/W | R/W | R/W | | | | R/W |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNIHEN | MTIHEN | NBC | CAF | JFRAME | DNVLAN | FD | SPD |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 30 – IRXER** Ignore IPG GRXER
When set, GRXER has no effect on the GMAC's operation when GRXDV is low.

**Bit 29 – RXBP** Receive Bad Preamble
When set, frames with non-standard preamble are not rejected.

**Bit 28 – IPGSEN** IP Stretch Enable
When set, the transmit IPG can be increased above 96 bit times depending on the previous frame length using the IPG Stretch Register.

**Bit 26 – IRXFCS** Ignore RX FCS
When set, frames with FCS/CRC errors will not be rejected. FCS error statistics will still be collected for frames with bad FCS and FCS status will be recorded in frame’s DMA descriptor. For normal operation this bit must be set to zero.

**Bit 25 – EFRHD** Enable Frames Received in Half Duplex
Enable frames to be received in half-duplex mode while transmitting.

**Bit 24 – RXCOEN** Receive Checksum Offload Enable
When set, the receive checksum engine is enabled. Frames with bad IP, TCP or UDP checksums are discarded.

**Bit 23 – DCPF** Disable Copy of Pause Frames
Set to one to prevent valid pause frames being copied to memory. When set, pause frames are not copied to memory regardless of the state of the Copy All Frames bit, whether a hash match is found or whether a type ID match is identified. If a destination address match is found, the pause frame will be copied to memory. Note that valid pause frames received will still increment pause statistics and pause the transmission of frames as required.

**Bits 22:21 – DBW[1:0]** Data Bus Width
Should always be written to '0'.

**Bits 20:18 – CLK[2:0]** MDC CLock Division
Set according to MCK speed. These three bits determine the number MCK will be divided by to generate Management Data Clock (MDC). For conformance with the 802.3 specification, MDC must not exceed 2.5 MHz (MDC is only active during MDIO read and write operations).

| Value | Name | Description |
|---|---|---|
| 0 | MCK_8 | MCK divided by 8 (MCK up to 20 MHz) |
| 1 | MCK_16 | MCK divided by 16 (MCK up to 40 MHz) |
| 2 | MCK_32 | MCK divided by 32 (MCK up to 80 MHz) |
| 3 | MCK_48 | MCK divided by 48 (MCK up to 120 MHz) |
| 4 | MCK_64 | MCK divided by 64 (MCK up to 160 MHz) |
| 5 | MCK_96 | MCK divided by 96 (MCK up to 240 MHz) |

**Bit 17 – RFCS** Remove FCS
Setting this bit will cause received frames to be written to memory without their frame check sequence (last 4 bytes). The frame length indicated will be reduced by four bytes in this mode.

**Bit 16 – LFERD** Length Field Error Frame Discard
Setting this bit causes frames with a measured length shorter than the extracted length field (as indicated by bytes 13 and 14 in a non-VLAN tagged frame) to be discarded. This only applies to frames with a length field less than 0x0600.

**Bits 15:14 – RXBUFO[1:0]** Receive Buffer Offset
Indicates the number of bytes by which the received data is offset from the start of the receive buffer

**Bit 13 – PEN** Pause Enable
When set, transmission will pause if a non-zero 802.3 classic pause frame is received and PFC has not been negotiated.

**Bit 12 – RTY** Retry Test
Must be set to zero for normal operation. If set to one the backoff between collisions will always be one slot time. Setting this bit to one helps test the too many retries condition. Also used in the pause frame tests to reduce the pause counter's decrement time from 512 bit times, to every GRXCK cycle.

**Bit 8 – MAXFS** 1536 Maximum Frame Size
Setting this bit means the GMAC will accept frames up to 1536 bytes in length. Normally the GMAC would reject any frame above 1518 bytes.

**Bit 7 – UNIHEN** Unicast Hash Enable
When set, unicast frames will be accepted when the 6-bit hash function of the destination address points to a bit that is set in the Hash Register.

**Bit 6 – MTIHEN** Multicast Hash Enable
When set, multicast frames will be accepted when the 6-bit hash function of the destination address points to a bit that is set in the Hash Register.

**Bit 5 – NBC** No Broadcast
When set to logic one, frames addressed to the broadcast address of all ones will not be accepted.

**Bit 4 – CAF** Copy All Frames
When set to logic one, all valid frames will be accepted.

**Bit 3 – JFRAME** Jumbo Frame Size
Set to one to enable jumbo frames up to 10240 bytes to be accepted. The default length is 10240 bytes.

**Bit 2 – DNVLAN** Discard Non-VLAN FRAMES
When set only VLAN tagged frames will be passed to the address matching logic.

**Bit 1 – FD** Full Duplex
If set to logic one, the transmit block ignores the state of collision and carrier sense and allows receive while transmitting.

**Bit 0 – SPD** Speed
Set to logic one to indicate 100 Mbps operation, logic zero for 10 Mbps.

### 39.8.3 GMAC Network Status Register

**Name:** GMAC_NSR
**Offset:** 0x008
**Reset:** 0x000001x0
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXLPIS | | | | | IDLE | MDIO | |
| Access | R | | | | | R | R | |
| Reset | 0 | | | | | 0 | 0 | |

**Bit 7 – RXLPIS** LPI Indication
Low power idle has been detected on receive. This bit is set when LPI is detected and reset when normal idle is detected. An interrupt is generated when the state of this bit changes.

**Bit 2 – IDLE** PHY Management Logic Idle
The PHY management logic is idle (i.e., has completed).

**Bit 1 – MDIO** MDIO Input Status
Returns status of the MDIO pin.

### 39.8.4 GMAC User Register

**Name:** GMAC_UR
**Offset:** 0x00C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | RMII |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – RMII** Reduced MII Mode

| Value | Description |
|---|---|
| 0 | MII mode is selected (default). |
| 1 | RMII mode is selected. |

### 39.8.5 GMAC DMA Configuration Register

**Name:** GMAC_DCFGR
**Offset:** 0x010
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DDRP |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DRBS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TXCOEN | TXPBMS | RXBMS[1:0] | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ESPA | ESMA | | FBDLO[4:0] | | | | |
| Access | R/W | R/W | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | | 0 | 0 | 0 | 0 | 0 |

**Bit 24 – DDRP** DMA Discard Receive Packets
When set, the GMAC DMA will automatically discard receive packets from the receiver packet buffer memory when no AHB resource is available.

When low, the received packets will remain to be stored in the SRAM based packet buffer until AHB buffer resource next becomes available.

A write to this bit is ignored if the DMA is not configured in the packet buffer full store and forward mode.

**Bits 23:16 – DRBS[7:0]** DMA Receive Buffer Size
DMA receive buffer size in AHB system memory. The value defined by these bits determines the size of buffer to use in main AHB system memory when writing received data.

The value is defined in multiples of 64 bytes, thus a value of 0x01 corresponds to buffers of 64 bytes, 0x02 corresponds to 128 bytes etc.

For example:

– 0x02: 128 bytes

– 0x18: 1536 bytes (1 × max length frame/buffer)

– 0xA0: 10240 bytes (1 × 10K jumbo frame/buffer)

Note that this value should never be written as zero.

**Bit 11 – TXCOEN** Transmitter Checksum Generation Offload Enable
Transmitter IP, TCP and UDP checksum generation offload enable. When set, the transmitter checksum generation engine is enabled to calculate and substitute checksums for transmit frames. When clear, frame data is unaffected.

**Bit 10 – TXPBMS** Transmitter Packet Buffer Memory Size Select
Having this bit at zero halves the amount of memory used for the transmit packet buffer. This reduces the amount of memory used by the GMAC. It is important to set this bit to one if the full configured physical memory is available. The value in brackets below represents the size that would result for the default maximum configured memory size of 4 Kbytes.

| Value | Description |
|---|---|
| 0 | Do not use top address bit (2 Kbytes). |
| 1 | Use full configured addressable space (4 Kbytes). |

**Bits 9:8 – RXBMS[1:0]** Receiver Packet Buffer Memory Size Select
The default receive packet buffer size is 4 Kbytes. The table below shows how to configure this memory to FULL, HALF, QUARTER or EIGHTH of the default size.

| Value | Name | Description |
|---|---|---|
| 0 | EIGHTH | 4/8 Kbyte Memory Size |
| 1 | QUARTER | 4/4 Kbytes Memory Size |
| 2 | HALF | 4/2 Kbytes Memory Size |
| 3 | FULL | 4 Kbytes Memory Size |

**Bit 7 – ESPA** Endian Swap Mode Enable for Packet Data Accesses
When set, selects swapped endianism for AHB transfers. When clear, selects little endian mode.

**Bit 6 – ESMA** Endian Swap Mode Enable for Management Descriptor Accesses
When set, selects swapped endianism for AHB transfers. When clear, selects little endian mode.

**Bits 4:0 – FBDLO[4:0]** Fixed Burst Length for DMA Data Operations
Selects the burst length to attempt to use on the AHB when transferring frame data. Not used for DMA management operations and only used where space and data size allow. Otherwise SINGLE type AHB transfers are used.
One-hot priority encoding enforced automatically on register writes as follows, where ‘x’ represents don’t care:

| Value | Name | Description |
|---|---|---|
| 0 | – | Reserved |
| 1 | SINGLE | 00001: Always use SINGLE AHB bursts |
| 2 | – | Reserved |
| 4 | INCR4 | 001xx: Attempt to use INCR4 AHB bursts (Default) |
| 8 | INCR8 | 01xxx: Attempt to use INCR8 AHB bursts |
| 16 | INCR16 | 1xxxx: Attempt to use INCR16 AHB bursts |

### 39.8.6 GMAC Transmit Status Register

**Name:** GMAC_TSR
**Offset:** 0x014
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | HRESP |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TXCOMP | TFC | TXGO | RLE | COL | UBR |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 8 – HRESP** HRESP Not OK
Set when the DMA block sees HRESP not OK. Writing a one clears this bit.

**Bit 5 – TXCOMP** Transmit Complete
Set when a frame has been transmitted. Writing a one clears this bit.

**Bit 4 – TFC** Transmit Frame Corruption Due to AHB Error
Transmit frame corruption due to AHB error. Set if an error occurs while midway through reading transmit frame from the AHB, including HRESP errors and buffers exhausted mid frame (if the buffers run out during transmission of a frame then transmission stops, FCS shall be bad and GTXER asserted).

Also set in DMA packet buffer mode if single frame is too large for configured packet buffer memory size.

Writing a one clears this bit.

**Bit 3 – TXGO** Transmit Go
Transmit go, if high transmit is active. When using the DMA interface, this bit represents the TXGO variable as specified in the transmit buffer description.

**Bit 2 – RLE** Retry Limit Exceeded
Writing a one clears this bit.

**Bit 1 – COL** Collision Occurred
Set by the assertion of collision. Writing a one clears this bit. When operating in 10/100 mode, this status indicates either a collision or a late collision.

**Bit 0 – UBR** Used Bit Read
Set when a transmit buffer descriptor is read with its used bit set. Writing a one clears this bit.

### 39.8.7 GMAC Receive Buffer Queue Base Address Register

**Name:** GMAC_RBQB
**Offset:** 0x018
**Reset:** 0x00000000
**Property:** Read/Write

This register holds the start address of the receive buffer queue (receive buffers descriptor list). The receive buffer queue base address must be initialized before receive is enabled through bit 2 of the Network Control Register. Once reception is enabled, any write to the Receive Buffer Queue Base Address Register is ignored. Reading this register returns the location of the descriptor currently being accessed. This value increments as buffers are used. Software should not use this register for determining where to remove received frames from the queue as it constantly changes as new frames are received. Software should instead work its way through the buffer descriptor queue checking the "used" bits.

In terms of AMBA AHB operation, the descriptors are read from memory using a single 32-bit AHB access. The descriptors should be aligned at 32-bit boundaries and the descriptors are written to using two individual non sequential accesses.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – ADDR[29:0]** Receive Buffer Queue Base Address
Written with the address of the start of the receive queue.

### 39.8.8 GMAC Transmit Buffer Queue Base Address Register

**Name:** GMAC_TBQB
**Offset:** 0x01C
**Reset:** 0x00000000
**Property:** Read/Write

This register holds the start address of the transmit buffer queue (transmit buffers descriptor list). The Transmit Buffer Queue Base Address Register must be initialized before transmit is started through bit 9 of the Network Control Register. Once transmission has started, any write to the Transmit Buffer Queue Base Address Register is illegal and therefore ignored.

Note that due to clock boundary synchronization, it takes a maximum of four MCK cycles from the writing of the transmit start bit before the transmitter is active. Writing to the Transmit Buffer Queue Base Address Register during this time may produce unpredictable results.

Reading this register returns the location of the descriptor currently being accessed. Since the DMA handles two frames at once, this may not necessarily be pointing to the current frame being transmitted.

In terms of AMBA AHB operation, the descriptors are written to memory using a single 32-bit AHB access. The descriptors should be aligned at 32-bit boundaries and the descriptors are read from memory using two individual non sequential accesses.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – ADDR[29:0]** Transmit Buffer Queue Base Address
Written with the address of the start of the transmit queue.

### 39.8.9 GMAC Receive Status Register

**Name:** GMAC_RSR
**Offset:** 0x020
**Reset:** 0x00000000
**Property:** Read/Write

This register, when read, provides receive status details. Once read, individual bits may be cleared by writing a one to them. It is not possible to set a bit to 1 by writing to the register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | HNO | RXOVR | REC | BNA |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 3 – HNO** HRESP Not OK
Set when the DMA block sees HRESP not OK. Writing a one clears this bit.

**Bit 2 – RXOVR** Receive Overrun
This bit is set if the receive status was not taken at the end of the frame. This bit is also set if the packet buffer overflows. The buffer will be recovered if an overrun occurs. Writing a one clears this bit.

**Bit 1 – REC** Frame Received
One or more frames have been received and placed in memory. Writing a one clears this bit.

**Bit 0 – BNA** Buffer Not Available
An attempt was made to get a new buffer and the pointer indicated that it was owned by the processor. The DMA will re-read the pointer each time an end of frame is received until a valid pointer is found. This bit is set following each descriptor read attempt that fails, even if consecutive pointers are unsuccessful and software has in the mean time cleared the status flag. Writing a one clears this bit.

### 39.8.10 GMAC Interrupt Status Register

**Name:** GMAC_ISR
**Offset:** 0x024
**Reset:** 0x00000000
**Property:** Read-only

This register indicates the source of the interrupt. In order that the bits of this register read 1, the corresponding interrupt source must be enabled in the mask register. If any bit is set in this register, the GMAC interrupt signal will be asserted in the system.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | TSUTIMCOMP | WOL | RXLPISBC | SRI | PDRSFT | PDRQFT |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PDRSFR | PDRQFR | SFT | DRQFT | SFR | DRQFR | | |
| Access | R | R | R | R | R | R | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | PFTR | PTZ | PFNZ | HRESP | ROVR | | |
| Access | | R | R | R | R | R | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCOMP | TFC | RLEX | TUR | TXUBR | RXUBR | RCOMP | MFS |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 29 – TSUTIMCOMP** TSU Timer Comparison
Indicates when TSU timer count value is equal to programmed value. Cleared on read.

**Bit 28 – WOL** Wake On LAN
WOL interrupt. Indicates a WOL event has been received.

**Bit 27 – RXLPISBC** Receive LPI indication Status Bit Change
Receive LPI indication status bit change. Cleared on read.

**Bit 26 – SRI** TSU Seconds Register Increment
Indicates the register has incremented. Cleared on read.

**Bit 25 – PDRSFT** PDelay Response Frame Transmitted
Indicates a PTP pdelay_resp frame has been transmitted. Cleared on read.

**Bit 24 – PDRQFT** PDelay Request Frame Transmitted
Indicates a PTP pdelay_req frame has been transmitted. Cleared on read.

**Bit 23 – PDRSFR** PDelay Response Frame Received
Indicates a PTP pdelay_resp frame has been received. Cleared on read.

**Bit 22 – PDRQFR** PDelay Request Frame Received
Indicates a PTP pdelay_req frame has been received. Cleared on read.

**Bit 21 – SFT** PTP Sync Frame Transmitted
Indicates a PTP sync frame has been transmitted. Cleared on read.

**Bit 20 – DRQFT** PTP Delay Request Frame Transmitted
Indicates a PTP delay_req frame has been transmitted. Cleared on read.

**Bit 19 – SFR** PTP Sync Frame Received
Indicates a PTP sync frame has been received. Cleared on read.

**Bit 18 – DRQFR** PTP Delay Request Frame Received
Indicates a PTP delay_req frame has been received. Cleared on read.

**Bit 14 – PFTR** Pause Frame Transmitted
Indicates a pause frame has been successfully transmitted after being initiated from the Network Control register. Cleared on read.

**Bit 13 – PTZ** Pause Time Zero
Set when either the Pause Time register at address 0x38 decrements to zero, or when a valid pause frame is received with a zero pause quantum field. Cleared on read.

**Bit 12 – PFNZ** Pause Frame with Non-zero Pause Quantum Received
Indicates a valid pause has been received that has a non-zero pause quantum field. Cleared on read.

**Bit 11 – HRESP** HRESP Not OK
Set when the DMA block sees HRESP not OK. Cleared on read.

**Bit 10 – ROVR** Receive Overrun
Set when the receive overrun status bit is set. Cleared on read.

**Bit 7 – TCOMP** Transmit Complete
Set when a frame has been transmitted. Cleared on read.

**Bit 6 – TFC** Transmit Frame Corruption Due to AHB Error
Transmit frame corruption due to AHB error. Set if an error occurs while midway through reading transmit frame from the AHB, including HRESP errors and buffers exhausted mid frame.

**Bit 5 – RLEX** Retry Limit Exceeded
Transmit error. Cleared on read.

**Bit 4 – TUR** Transmit Underrun
This interrupt is set if the transmitter was forced to terminate a frame that it has already began transmitting due to further data being unavailable.

This interrupt is set if a transmitter status write back has not completed when another status write back is attempted.

This interrupt is also set when the transmit DMA has written the SOP data into the FIFO and either the AHB bus was not granted in time for further data, or because an AHB not OK response was returned, or because the used bit was read.

**Bit 3 – TXUBR** TX Used Bit Read
Set when a transmit buffer descriptor is read with its used bit set. Cleared on read.

**Bit 2 – RXUBR** RX Used Bit Read
Set when a receive buffer descriptor is read with its used bit set. Cleared on read.

**Bit 1 – RCOMP** Receive Complete
A frame has been stored in memory. Cleared on read.

**Bit 0 – MFS** Management Frame Sent
The PHY Maintenance Register has completed its operation. Cleared on read.

© 2018 Microchip Technology Inc.

### 39.8.11 GMAC Interrupt Enable Register

**Name:** GMAC_IER
**Offset:** 0x028
**Reset:** –
**Property:** Write-only

This register is write-only and when read will return zero.

The following values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | TSUTIMCOMP | WOL | RXLPISBC | SRI | PDRSFT | PDRQFT |
| Access | | | W | W | W | W | W | W |
| Reset | | | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PDRSFR | PDRQFR | SFT | DRQFT | SFR | DRQFR | | |
| Access | W | W | W | W | W | W | | |
| Reset | – | – | – | – | – | – | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EXINT | PFTR | PTZ | PFNZ | HRESP | ROVR | | |
| Access | W | W | W | W | W | W | | |
| Reset | – | – | – | – | – | – | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCOMP | TFC | RLEX | TUR | TXUBR | RXUBR | RCOMP | MFS |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 29 – TSUTIMCOMP** TSU Timer Comparison

**Bit 28 – WOL** Wake On LAN

**Bit 27 – RXLPISBC** Enable RX LPI Indication

**Bit 26 – SRI** TSU Seconds Register Increment

**Bit 25 – PDRSFT** PDelay Response Frame Transmitted

**Bit 24 – PDRQFT** PDelay Request Frame Transmitted

**Bit 23 – PDRSFR** PDelay Response Frame Received

**Bit 22 – PDRQFR** PDelay Request Frame Received

**Bit 21 – SFT** PTP Sync Frame Transmitted

**Bit 20 – DRQFT** PTP Delay Request Frame Transmitted

**Bit 19 – SFR** PTP Sync Frame Received

**Bit 18 – DRQFR** PTP Delay Request Frame Received

**Bit 15 – EXINT** External Interrupt

**Bit 14 – PFTR** Pause Frame Transmitted

**Bit 13 – PTZ** Pause Time Zero

**Bit 12 – PFNZ** Pause Frame with Non-zero Pause Quantum Received

**Bit 11 – HRESP** HRESP Not OK

**Bit 10 – ROVR** Receive Overrun

**Bit 7 – TCOMP** Transmit Complete

**Bit 6 – TFC** Transmit Frame Corruption Due to AHB Error

**Bit 5 – RLEX** Retry Limit Exceeded or Late Collision

**Bit 4 – TUR** Transmit Underrun

**Bit 3 – TXUBR** TX Used Bit Read

**Bit 2 – RXUBR** RX Used Bit Read

**Bit 1 – RCOMP** Receive Complete

**Bit 0 – MFS** Management Frame Sent

### 39.8.12 GMAC Interrupt Disable Register

**Name:** GMAC_IDR
**Offset:** 0x02C
**Reset:** –
**Property:** Write-only

This register is write-only and when read will return zero.

The following values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | TSUTIMCOMP | WOL | RXLPISBC | SRI | PDRSFT | PDRQFT |
| Access | | | W | W | W | W | W | W |
| Reset | | | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PDRSFR | PDRQFR | SFT | DRQFT | SFR | DRQFR | | |
| Access | W | W | W | W | W | W | | |
| Reset | – | – | – | – | – | – | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EXINT | PFTR | PTZ | PFNZ | HRESP | ROVR | | |
| Access | W | W | W | W | W | W | | |
| Reset | – | – | – | – | – | – | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCOMP | TFC | RLEX | TUR | TXUBR | RXUBR | RCOMP | MFS |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 29 – TSUTIMCOMP** TSU Timer Comparison

**Bit 28 – WOL** Wake On LAN

**Bit 27 – RXLPISBC** Enable RX LPI Indication

**Bit 26 – SRI** TSU Seconds Register Increment

**Bit 25 – PDRSFT** PDelay Response Frame Transmitted

**Bit 24 – PDRQFT** PDelay Request Frame Transmitted

**Bit 23 – PDRSFR** PDelay Response Frame Received

**Bit 22 – PDRQFR** PDelay Request Frame Received

**Bit 21 – SFT** PTP Sync Frame Transmitted

**Bit 20 – DRQFT** PTP Delay Request Frame Transmitted

**Bit 19 – SFR** PTP Sync Frame Received

**Bit 18 – DRQFR** PTP Delay Request Frame Received

**Bit 15 – EXINT** External Interrupt

**Bit 14 – PFTR** Pause Frame Transmitted

**Bit 13 – PTZ** Pause Time Zero

**Bit 12 – PFNZ** Pause Frame with Non-zero Pause Quantum Received

**Bit 11 – HRESP** HRESP Not OK

**Bit 10 – ROVR** Receive Overrun

**Bit 7 – TCOMP** Transmit Complete

**Bit 6 – TFC** Transmit Frame Corruption Due to AHB Error

**Bit 5 – RLEX** Retry Limit Exceeded or Late Collision

**Bit 4 – TUR** Transmit Underrun

**Bit 3 – TXUBR** TX Used Bit Read

**Bit 2 – RXUBR** RX Used Bit Read

**Bit 1 – RCOMP** Receive Complete

**Bit 0 – MFS** Management Frame Sent

### 39.8.13 GMAC Interrupt Mask Register

**Name:** GMAC_IMR
**Offset:** 0x030
**Reset:** 0x07FFFFFF
**Property:** Read/Write

The Interrupt Mask Register is a read-only register indicating which interrupts are masked. All bits are set at reset and can be reset individually by writing to the Interrupt Enable Register or set individually by writing to the Interrupt Disable Register. Having separate address locations for enable and disable saves the need for performing a read modify write when updating the Interrupt Mask Register.

For test purposes there is a write-only function to this register that allows the bits in the Interrupt Status Register to be set or cleared, regardless of the state of the mask register. A write to this register directly affects the state of the corresponding bit in the Interrupt Status Register, causing an interrupt to be generated if a 1 is written.

The following values are valid for all listed bit names of this register when read:

0: The corresponding interrupt is enabled.

1: The corresponding interrupt is not enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | TSUTIMCOMP | WOL | RXLPISBC | SRI | PDRSFT | PDRQFT |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 1 | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PDRSFR | PDRQFR | SFT | DRQFT | SFR | DRQFR | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EXINT | PFTR | PTZ | PFNZ | HRESP | ROVR | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCOMP | TFC | RLEX | TUR | TXUBR | RXUBR | RCOMP | MFS |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bit 29 – TSUTIMCOMP** TSU Timer Comparison

**Bit 28 – WOL** Wake On LAN

**Bit 27 – RXLPISBC** Enable RX LPI Indication

**Bit 26 – SRI** TSU Seconds Register Increment

**Bit 25 – PDRSFT** PDelay Response Frame Transmitted

**Bit 24 – PDRQFT** PDelay Request Frame Transmitted

**Bit 23 – PDRSFR** PDelay Response Frame Received

**Bit 22 – PDRQFR** PDelay Request Frame Received

**Bit 21 – SFT** PTP Sync Frame Transmitted

**Bit 20 – DRQFT** PTP Delay Request Frame Transmitted

**Bit 19 – SFR** PTP Sync Frame Received

**Bit 18 – DRQFR** PTP Delay Request Frame Received

**Bit 15 – EXINT** External Interrupt

**Bit 14 – PFTR** Pause Frame Transmitted

**Bit 13 – PTZ** Pause Time Zero

**Bit 12 – PFNZ** Pause Frame with Non-zero Pause Quantum Received

**Bit 11 – HRESP** HRESP Not OK

**Bit 10 – ROVR** Receive Overrun

**Bit 7 – TCOMP** Transmit Complete

**Bit 6 – TFC** Transmit Frame Corruption Due to AHB Error

**Bit 5 – RLEX** Retry Limit Exceeded

**Bit 4 – TUR** Transmit Underrun

**Bit 3 – TXUBR** TX Used Bit Read

**Bit 2 – RXUBR** RX Used Bit Read

**Bit 1 – RCOMP** Receive Complete

**Bit 0 – MFS** Management Frame Sent

### 39.8.14 GMAC PHY Maintenance Register

**Name:** GMAC_MAN
**Offset:** 0x034
**Reset:** 0x00000000
**Property:** Read/Write

The PHY Maintenance Register is implemented as a shift register. Writing to the register starts a shift operation which is signalled as complete when bit 2 is set in the Network Status Register. It takes about 2000 MCK cycles to complete, when MDC is set for MCK divide by 32 in the Network Configuration Register. An interrupt is generated upon completion.

During this time, the MSB of the register is output on the MDIO pin and the LSB updated from the MDIO pin with each MDC cycle. This causes transmission of a PHY management frame on MDIO. See Section 22.2.4.5 of the IEEE 802.3 standard.

Reading during the shift operation returns the current contents of the shift register. At the end of management operation, the bits will have shifted back to their original locations. For a read operation, the data bits are updated with data read from the PHY. It is important to write the correct values to the register to ensure a valid PHY management frame is produced.

The MDIO interface can read IEEE 802.3 clause 45 PHYs as well as clause 22 PHYs. To read clause 45 PHYs, bit 30 should be written with a 0 rather than a 1. To write clause 45 PHYs, bits 31:28 should be written as 0x0001. See the table below.

**Table 39-16. Clause 22/Clause 45 PHYs Read/Write Access Configuration (GMAC_MAN Bits 31:28)**

| PHY | Access | Bit Value | | | |
|---|---|---|---|---|---|
| | | WZO | CLTTO | OP[1] | OP[0] |
| Clause 22 | Read | 0 | 1 | 1 | 0 |
| | Write | 0 | 1 | 0 | 1 |
| Clause 45 | Read | 0 | 0 | 1 | 1 |
| | Write | 0 | 0 | 0 | 1 |
| | Read + Address | 0 | 0 | 1 | 0 |

For a description of MDC generation, see GMAC Network Configuration Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WZO | CLTTO | OP[1:0] | | PHYA[4:1] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PHYA[0:0] | REGA[4:0] | | | | | WTN[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – WZO** Write ZERO
Must be written with 0.

**Bit 30 – CLTTO** Clause 22 Operation

| Value | Description |
|---|---|
| 0 | Clause 45 operation |
| 1 | Clause 22 operation |

**Bits 29:28 – OP[1:0]** Operation

| Value | Description |
|---|---|
| 01 | Write |
| 10 | Read |

**Bits 27:23 – PHYA[4:0]** PHY Address

**Bits 22:18 – REGA[4:0]** Register Address
Specifies the register in the PHY to access.

**Bits 17:16 – WTN[1:0]** Write Ten
Must be written to 10.

**Bits 15:0 – DATA[15:0]** PHY Data
For a write operation this field is written with the data to be written to the PHY. After a read operation this field contains the data read from the PHY.

### 39.8.15 GMAC Receive Pause Quantum Register

**Name:** GMAC_RPQ
**Offset:** 0x038
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RPQ[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RPQ[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RPQ[15:0]** Received Pause Quantum
Stores the current value of the Receive Pause Quantum Register which is decremented every 512 bit times.

### 39.8.16 GMAC Transmit Pause Quantum Register

**Name:** GMAC_TPQ
**Offset:** 0x03C
**Reset:** 0x0000FFFF
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TPQ[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TPQ[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 15:0 – TPQ[15:0]** Transmit Pause Quantum
Written with the pause quantum value for pause frame transmission.

### 39.8.17 GMAC TX Partial Store and Forward Register

**Name:** GMAC_TPSF
**Offset:** 0x040
**Reset:** 0x00000FFF
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ENTXP | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TPB1ADR[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TPB1ADR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bit 31 – ENTXP** Enable TX Partial Store and Forward Operation

**Bits 11:0 – TPB1ADR[11:0]** Transmit Partial Store and Forward Address Watermark value. Reset = 1.

### 39.8.18 GMAC RX Partial Store and Forward Register

**Name:** GMAC_RPSF
**Offset:** 0x044
**Reset:** 0x00000FFF
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ENRXP | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RPB1ADR[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RPB1ADR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bit 31 – ENRXP** Enable RX Partial Store and Forward Operation

**Bits 11:0 – RPB1ADR[11:0]** Receive Partial Store and Forward Address Watermark value. Reset = 1.

### 39.8.19 GMAC RX Jumbo Frame Max Length Register

**Name:** GMAC_RJFML
**Offset:** 0x048
**Reset:** 0x00003FFF
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | FML[13:8] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FML[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 13:0 – FML[13:0]** Frame Max Length
Rx jumbo frame maximum length.

### 39.8.20 GMAC Hash Register Bottom

**Name:** GMAC_HRB
**Offset:** 0x080
**Reset:** 0x00000000
**Property:** Read/Write

The unicast hash enable (UNIHEN) and the multicast hash enable (MITIHEN) bits in the Network Configuration Register (GMAC Network Configuration Register) enable the reception of hash matched frames. See Hash Addressing.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** Hash Address
The first 32 bits of the Hash Address Register.

### 39.8.21 GMAC Hash Register Top

**Name:** GMAC_HRT
**Offset:** 0x084
**Reset:** 0x00000000
**Property:** Read/Write

The unicast hash enable (UNIHEN) and the multicast hash enable (MITIHEN) bits in the GMAC Network Configuration Register enable the reception of hash matched frames. See Hash Addressing.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** Hash Address
Bits 63 to 32 of the Hash Address Register.

### 39.8.22 GMAC Specific Address 1 Bottom Register

**Name:** GMAC_SAB1
**Offset:** 0x088
**Reset:** 0x00000000
**Property:** Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** Specific Address 1
Least significant 32 bits of the destination address, that is, bits 31:0. Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

### 39.8.23 GMAC Specific Address 1 Top Register

**Name:** GMAC_SAT1
**Offset:** 0x08C
**Reset:** 0x00000000
**Property:** Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – ADDR[15:0]** Specific Address 1
The most significant bits of the destination address, that is, bits 47:32.

### 39.8.24 GMAC Specific Address 2 Bottom Register

**Name:** GMAC_SAB2
**Offset:** 0x090
**Reset:** 0x00000000
**Property:** Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** Specific Address 2
Least significant 32 bits of the destination address, that is, bits 31:0. Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

### 39.8.25 GMAC Specific Address 2 Top Register

**Name:** GMAC_SAT2
**Offset:** 0x094
**Reset:** 0x00000000
**Property:** Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – ADDR[15:0]** Specific Address 2
The most significant bits of the destination address, that is, bits 47:32.

### 39.8.26 GMAC Specific Address 3 Bottom Register

**Name:** GMAC_SAB3
**Offset:** 0x098
**Reset:** 0x00000000
**Property:** Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** Specific Address 3
Least significant 32 bits of the destination address, that is, bits 31:0. Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

### 39.8.27 GMAC Specific Address 3 Top Register

**Name:** GMAC_SAT3
**Offset:** 0x09C
**Reset:** 0x00000000
**Property:** Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – ADDR[15:0]** Specific Address 3
The most significant bits of the destination address, that is, bits 47:32.

### 39.8.28 GMAC Specific Address 4 Bottom Register

**Name:** GMAC_SAB4
**Offset:** 0x0A0
**Reset:** 0x00000000
**Property:** Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** Specific Address 4
Least significant 32 bits of the destination address, that is, bits 31:0. Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

### 39.8.29 GMAC Specific Address 4 Top Register

**Name:** GMAC_SAT4
**Offset:** 0x0A4
**Reset:** 0x00000000
**Property:** Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – ADDR[15:0]** Specific Address 4
The most significant bits of the destination address, that is, bits 47:32.

### 39.8.30 GMAC Type ID Match 1 Register

**Name:** GMAC_TIDM1
**Offset:** 0x0A8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ENID1 | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TID[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TID[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – ENID1** Enable Copying of TID Matched Frames

| Value | Description |
|---|---|
| 0 | TID is not part of the comparison match. |
| 1 | TID is processed for the comparison match. |

**Bits 15:0 – TID[15:0]** Type ID Match 1
For use in comparisons with received frames type ID/length frames.

### 39.8.31 GMAC Type ID Match 2 Register

**Name:** GMAC_TIDM2
**Offset:** 0x0AC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ENID2 | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TID[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TID[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – ENID2** Enable Copying of TID Matched Frames

| Value | Description |
|---|---|
| 0 | TID is not part of the comparison match. |
| 1 | TID is processed for the comparison match. |

**Bits 15:0 – TID[15:0]** Type ID Match 2
For use in comparisons with received frames type ID/length frames.

### 39.8.32 GMAC Type ID Match 3 Register

**Name:** GMAC_TIDM3
**Offset:** 0x0B0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ENID3 | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TID[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TID[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – ENID3** Enable Copying of TID Matched Frames

| Value | Description |
|---|---|
| 0 | TID is not part of the comparison match. |
| 1 | TID is processed for the comparison match. |

**Bits 15:0 – TID[15:0]** Type ID Match 3
For use in comparisons with received frames type ID/length frames.

### 39.8.33 GMAC Type ID Match 4 Register

**Name:** GMAC_TIDM4
**Offset:** 0x0B4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ENID4 | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TID[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TID[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – ENID4** Enable Copying of TID Matched Frames

| Value | Description |
|---|---|
| 0 | TID is not part of the comparison match. |
| 1 | TID is processed for the comparison match. |

**Bits 15:0 – TID[15:0]** Type ID Match 4
For use in comparisons with received frames type ID/length frames.

### 39.8.34 GMAC Wake on LAN Register

**Name:** GMAC_WOL
**Offset:** 0x0B8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MTI | SA1 | ARP | MAG |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IP[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IP[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 19 – MTI** Multicast Hash Event Enable
Wake on LAN multicast hash event enable.

**Bit 18 – SA1** Specific Address Register 1 Event Enable
Wake on LAN Specific Address Register 1 event enable.

**Bit 17 – ARP** ARP Request Event Enable
Wake on LAN ARP request event enable.

**Bit 16 – MAG** Magic Packet Event Enable
Wake on LAN magic packet event enable.

**Bits 15:0 – IP[15:0]** ARP Request IP Address
Wake on LAN ARP request IP address. Written to define the least significant 16 bits of the target IP address that is matched to generate a Wake on LAN event. A value of zero will not generate an event, even if this is matched by the received frame.

### 39.8.35 GMAC IPG Stretch Register

**Name:** GMAC_IPGS
**Offset:** 0x0BC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

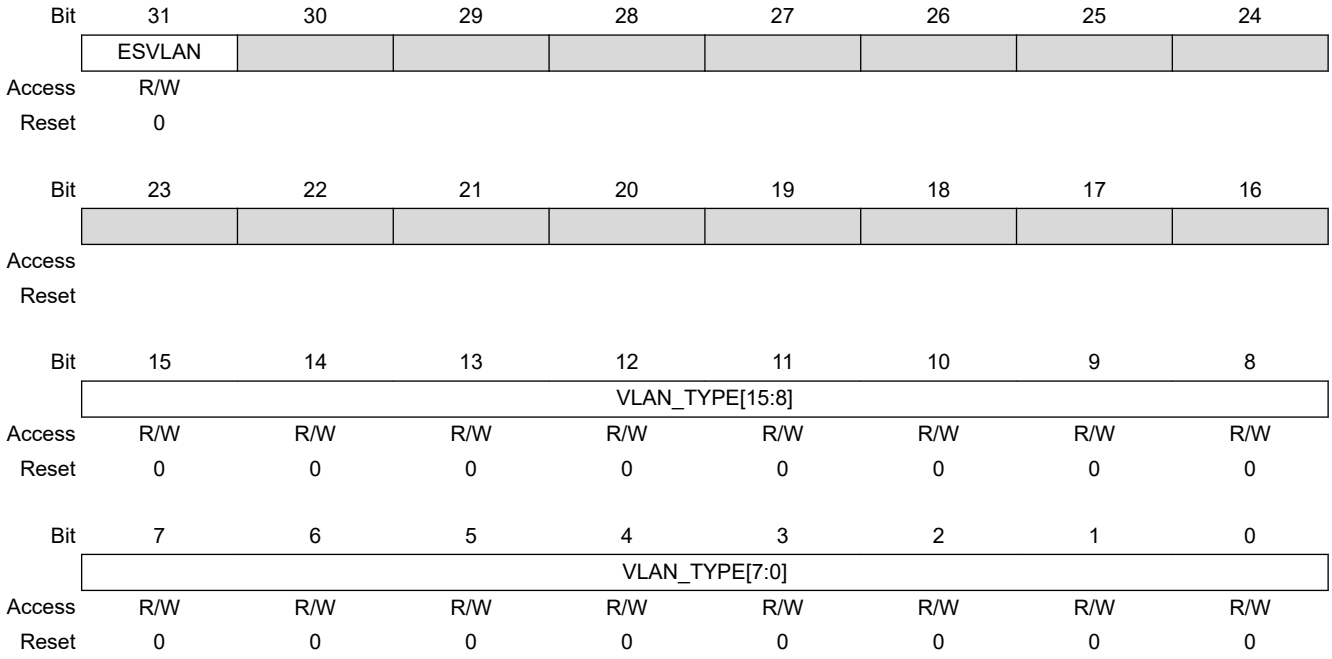
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FL[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – FL[15:0]** Frame Length
Bits 7:0 are multiplied with the previously transmitted frame length (including preamble). Bits 15:8 +1 divide the frame length. If the resulting number is greater than 96 and bit 28 is set in the Network Configuration Register then the resulting number is used for the transmit inter-packet-gap. 1 is added to bits 15:8 to prevent a divide by zero. See MAC Transmit Block.

### 39.8.36 GMAC Stacked VLAN Register

**Name:** GMAC_SVLAN
**Offset:** 0x0C0
**Reset:** 0x00000000
**Property:** Read/Write



| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ESVLAN | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | VLAN_TYPE[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | VLAN_TYPE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – ESVLAN** Enable Stacked VLAN Processing Mode

| Value | Description |
|---|---|
| 0 | Disable the stacked VLAN processing mode |
| 1 | Enable the stacked VLAN processing mode |

**Bits 15:0 – VLAN_TYPE[15:0]** User Defined VLAN_TYPE Field
User defined VLAN_TYPE field. When Stacked VLAN is enabled, the first VLAN tag in a received frame will only be accepted if the VLAN type field is equal to this user defined VLAN_TYPE, OR equal to the standard VLAN type (0x8100). Note that the second VLAN tag of a Stacked VLAN packet will only be matched correctly if its VLAN_TYPE field equals 0x8100.

### 39.8.37 GMAC Transmit PFC Pause Register

**Name:** GMAC_TPFCP
**Offset:** 0x0C4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PQ[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PEV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:8 – PQ[7:0]** Pause Quantum
If bit 17 of the Network Control Register is written with a one then for each entry equal to zero in the Transmit PFC Pause Register[15:8], the PFC pause frame's pause quantum field associated with that entry will be taken from the Transmit Pause Quantum Register. For each entry equal to one in the Transmit PFC Pause Register [15:8], the pause quantum associated with that entry will be zero.

**Bits 7:0 – PEV[7:0]** Priority Enable Vector
If bit 17 of the Network Control Register is written with a one then the priority enable vector of the PFC priority based pause frame will be set equal to the value stored in this register [7:0].

### 39.8.38 GMAC Specific Address 1 Mask Bottom Register

**Name:** GMAC_SAMB1
**Offset:** 0x0C8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** Specific Address 1 Mask
Setting a bit to one masks the corresponding bit in the Specific Address 1 Register.

### 39.8.39 GMAC Specific Address Mask 1 Top Register

**Name:** GMAC_SAMT1
**Offset:** 0x0CC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – ADDR[15:0]** Specific Address 1 Mask
Setting a bit to one masks the corresponding bit in the Specific Address 1 Register.

### 39.8.40 GMAC 1588 Timer Nanosecond Comparison Register

**Name:** GMAC_NSC
**Offset:** 0x0DC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | NANOSEC[21:16] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NANOSEC[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NANOSEC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 21:0 – NANOSEC[21:0]** 1588 Timer Nanosecond Comparison Value
Value is compared to the bits [45:24] of the TSU timer count value (upper 22 bits of nanosecond value).

### 39.8.41 GMAC 1588 Timer Second Comparison Low Register

**Name:** GMAC_SCL
**Offset:** 0x0E0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SEC[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SEC[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SEC[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SEC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – SEC[31:0]** 1588 Timer Second Comparison Value
Value is compared to seconds value bits [31:0] of the TSU timer count value.

### 39.8.42 GMAC 1588 Timer Second Comparison High Register

**Name:** GMAC_SCH
**Offset:** 0x0E4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

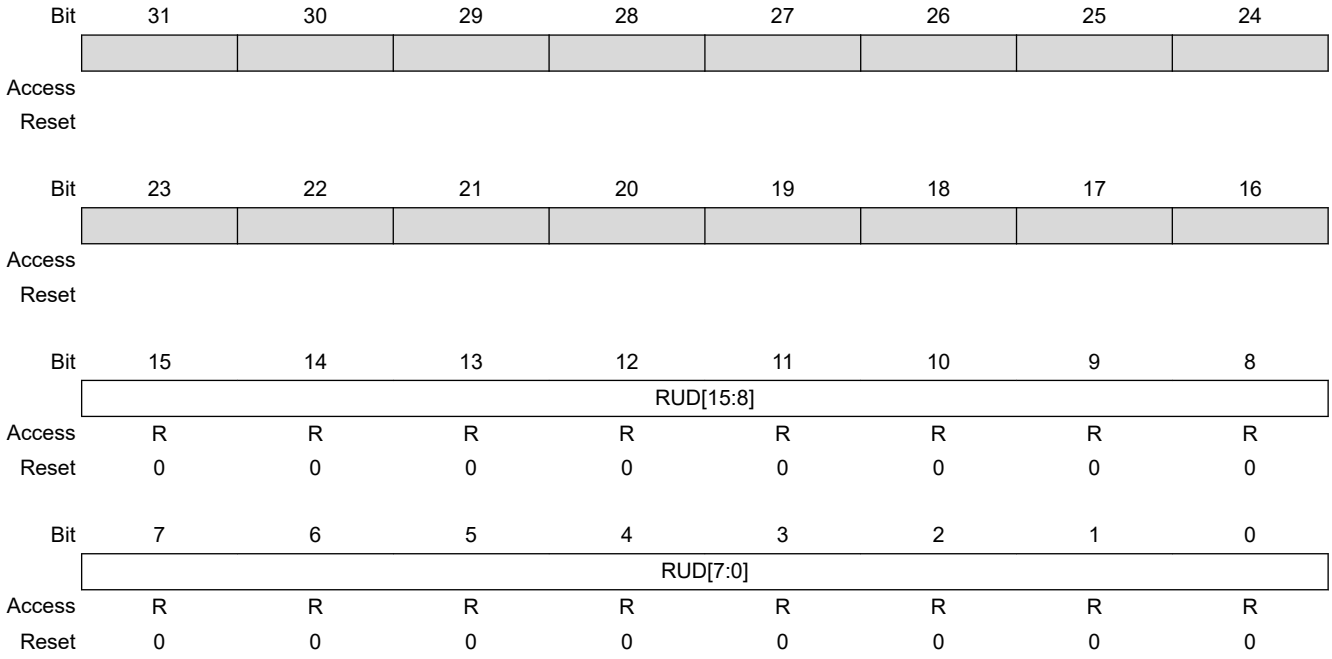
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SEC[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SEC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – SEC[15:0]** 1588 Timer Second Comparison Value
Value is compared to the top 16 bits (most significant 16 bits [47:32] of seconds value) of the TSU timer count value.

### 39.8.43 GMAC PTP Event Frame Transmitted Seconds High Register

**Name:** GMAC_EFTSH
**Offset:** 0x0FC
**Reset:** 0x00000000
**Property:** Read-only



| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RUD[15:0]** Register Update
The register is updated with the value that the 1588 timer seconds register held when the SFD of a PTP transmit primary event crosses the MII interface. An interrupt is issued when the register is updated.

### 39.8.44 GMAC PTP Event Frame Received Seconds High Register

**Name:** GMAC_EFRSH
**Offset:** 0x0EC
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RUD[15:0]** Register Update
The register is updated with the value that the 1588 timer seconds register held when the SFD of a PTP transmit primary event crosses the MII interface. An interrupt is issued when the register is updated.

### 39.8.45 GMAC PTP Peer Event Frame Transmitted Seconds High Register

**Name:** GMAC_PEFTSH
**Offset:** 0x0F0
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

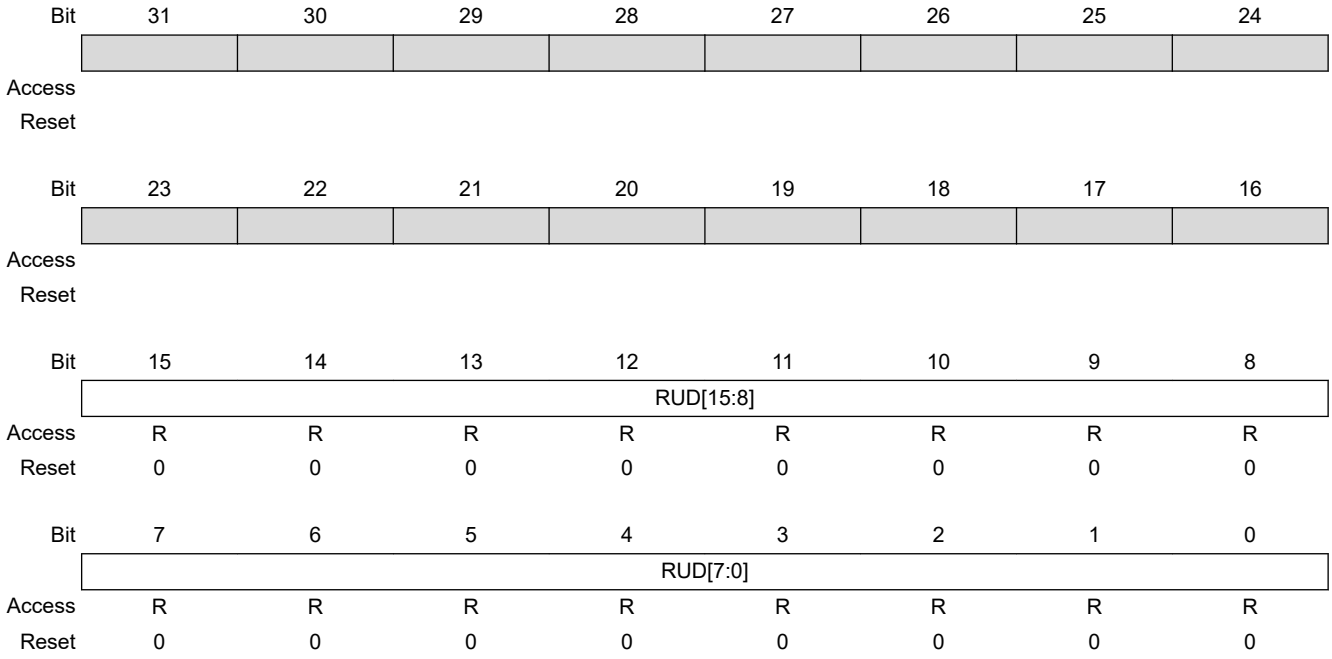
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RUD[15:0]** Register Update
The register is updated with the value that the 1588 timer seconds register held when the SFD of a PTP transmit peer event crosses the MII interface. An interrupt is issued when the register is updated.

### 39.8.46 GMAC PTP Peer Event Frame Received Seconds High Register

**Name:** GMAC_PEFRSH
**Offset:** 0x0F4
**Reset:** 0x00000000
**Property:** Read-only



| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RUD[15:0]** Register Update
The register is updated with the value that the 1588 timer seconds register held when the SFD of a PTP transmit peer event crosses the MII interface. An interrupt is issued when the register is updated.

### 39.8.47 GMAC Octets Transmitted Low Register

**Name:** GMAC_OTLO
**Offset:** 0x100
**Reset:** 0x00000000
**Property:** Read-only

When reading the Octets Transmitted and Octets Received Registers, bits 31:0 should be read prior to bits 47:32 to ensure reliable operation.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TXO[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TXO[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TXO[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXO[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – TXO[31:0]** Transmitted Octets
Transmitted octets in frame without errors [31:0]. The number of octets transmitted in valid frames of any type. This counter is 48-bits, and is read through two registers. This count does not include octets from automatically generated pause frames.

### 39.8.48 GMAC Octets Transmitted High Register

**Name:** GMAC_OTHI
**Offset:** 0x104
**Reset:** 0x00000000
**Property:** Read-only

When reading the Octets Transmitted and Octets Received Registers, bits 31:0 should be read prior to bits 47:32 to ensure reliable operation.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TXO[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXO[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – TXO[15:0]** Transmitted Octets
Transmitted octets in frame without errors [47:32]. The number of octets transmitted in valid frames of any type. This counter is 48-bits, and is read through two registers. This count does not include octets from automatically generated pause frames.

### 39.8.49 GMAC Frames Transmitted Register

**Name:** GMAC_FT
**Offset:** 0x108
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – FTX[31:0]** Frames Transmitted without Error
Frames transmitted without error. This register counts the number of frames successfully transmitted, i.e., no underrun and not too many retries. Excludes pause frames.

### 39.8.50 GMAC Broadcast Frames Transmitted Register

**Name:** GMAC_BCFT
**Offset:** 0x10C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – BFTX[31:0]** Broadcast Frames Transmitted without Error
Broadcast frames transmitted without error. This register counts the number of broadcast frames successfully transmitted without error, i.e., no underrun and not too many retries. Excludes pause frames.

### 39.8.51 GMAC Multicast Frames Transmitted Register

**Name:** GMAC_MFT
**Offset:** 0x110
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | MFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – MFTX[31:0]** Multicast Frames Transmitted without Error
This register counts the number of multicast frames successfully transmitted without error, i.e., no underrun and not too many retries. Excludes pause frames.

### 39.8.52 GMAC Pause Frames Transmitted Register

**Name:** GMAC_PFT
**Offset:** 0x114
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – PFTX[15:0]** Pause Frames Transmitted Register
This register counts the number of pause frames transmitted. Only pause frames triggered by the register interface or through the external pause pins are counted as pause frames. Pause frames received through the FIFO interface are counted in the frames transmitted counter.

### 39.8.53 GMAC 64 Byte Frames Transmitted Register

**Name:** GMAC_BFT64
**Offset:** 0x118
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFTX[31:0]** 64 Byte Frames Transmitted without Error
This register counts the number of 64 byte frames successfully transmitted without error, i.e., no underrun and not too many retries. Excludes pause frames.

### 39.8.54 GMAC 65 to 127 Byte Frames Transmitted Register

**Name:** GMAC_TBFT127
**Offset:** 0x11C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFTX[31:0]** 65 to 127 Byte Frames Transmitted without Error
This register counts the number of 65 to 127 byte frames successfully transmitted without error, i.e., no underrun and not too many retries. Excludes pause frames.

### 39.8.55 GMAC 128 to 255 Byte Frames Transmitted Register

**Name:** GMAC_TBFT255
**Offset:** 0x120
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFTX[31:0]** 128 to 255 Byte Frames Transmitted without Error
This register counts the number of 128 to 255 byte frames successfully transmitted without error, i.e., no underrun and not too many retries.

### 39.8.56 GMAC 256 to 511 Byte Frames Transmitted Register

**Name:** GMAC_TBFT511
**Offset:** 0x124
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFTX[31:0]** 256 to 511 Byte Frames Transmitted without Error
This register counts the number of 256 to 511 byte frames successfully transmitted without error, i.e., no underrun and not too many retries.

### 39.8.57 GMAC 512 to 1023 Byte Frames Transmitted Register

**Name:** GMAC_TBFT1023
**Offset:** 0x128
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFTX[31:0]** 512 to 1023 Byte Frames Transmitted without Error
This register counts the number of 512 to 1023 byte frames successfully transmitted without error, i.e., no underrun and not too many retries.

### 39.8.58 GMAC 1024 to 1518 Byte Frames Transmitted Register

**Name:** GMAC_TBFT1518
**Offset:** 0x12C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFTX[31:0]** 1024 to 1518 Byte Frames Transmitted without Error
This register counts the number of 1024 to 1518 byte frames successfully transmitted without error, i.e., no underrun and not too many retries.

### 39.8.59 GMAC Greater Than 1518 Byte Frames Transmitted Register

**Name:** GMAC_GTBFT1518
**Offset:** 0x130
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

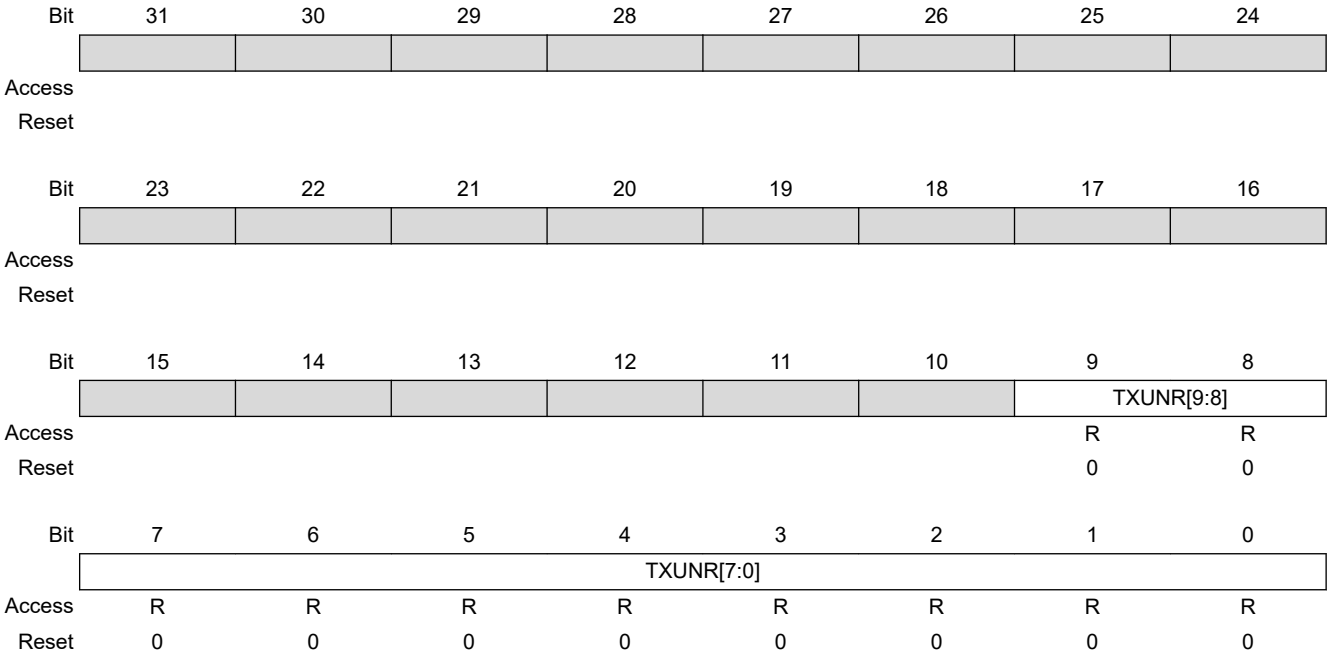
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFTX[31:0]** Greater than 1518 Byte Frames Transmitted without Error
This register counts the number of 1518 or above byte frames successfully transmitted without error i.e., no underrun and not too many retries.

### 39.8.60 GMAC Transmit Underruns Register

**Name:** GMAC_TUR
**Offset:** 0x134
**Reset:** 0x00000000
**Property:** Read-only




**Bits 9:0 – TXUNR[9:0]** Transmit Underruns
This register counts the number of frames not transmitted due to a transmit underrun. If this register is incremented then no other statistics register is incremented.

### 39.8.61 GMAC Single Collision Frames Register

**Name:** GMAC_SCF
**Offset:** 0x138
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | SCOL[17:16] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SCOL[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SCOL[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 17:0 – SCOL[17:0]** Single Collision
This register counts the number of frames experiencing a single collision before being successfully transmitted i.e., no underrun.

### 39.8.62 GMAC Multiple Collision Frames Register

**Name:** GMAC_MCF
**Offset:** 0x13C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | MCOL[17:16] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MCOL[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MCOL[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 17:0 – MCOL[17:0]** Multiple Collision
This register counts the number of frames experiencing between two and fifteen collisions prior to being successfully transmitted, i.e., no underrun and not too many retries.

### 39.8.63 GMAC Excessive Collisions Register

**Name:** GMAC_EC
**Offset:** 0x140
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

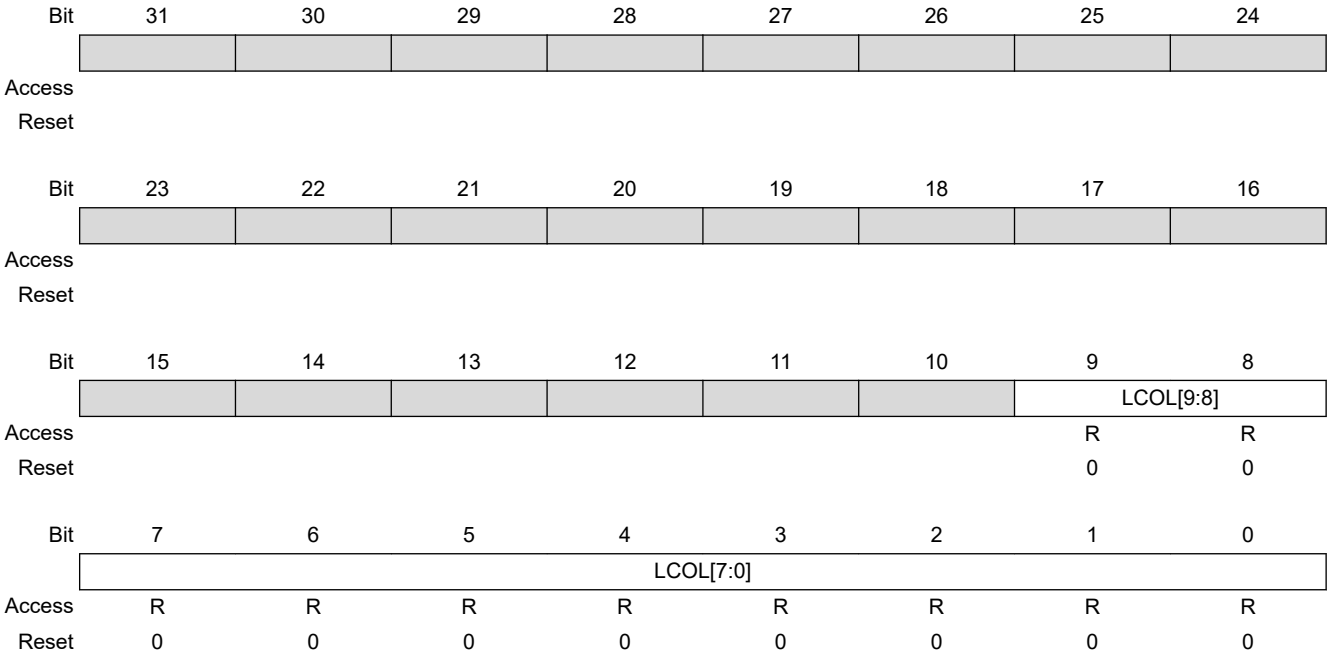
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | XCOL[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XCOL[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – XCOL[9:0]** Excessive Collisions
This register counts the number of frames that failed to be transmitted because they experienced 16 collisions.

### 39.8.64 GMAC Late Collisions Register

**Name:** GMAC_LC
**Offset:** 0x144
**Reset:** 0x00000000
**Property:** Read-only




**Bits 9:0 – LCOL[9:0]** Late Collisions
This register counts the number of late collisions occurring after the slot time (512 bits) has expired. In 10/100 mode, late collisions are counted twice i.e., both as a collision and a late collision.

### 39.8.65 GMAC Deferred Transmission Frames Register

**Name:** GMAC_DTF
**Offset:** 0x148
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | DEFT[17:16] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DEFT[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DEFT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 17:0 – DEFT[17:0]** Deferred Transmission
This register counts the number of frames experiencing deferral due to carrier sense being active on their first attempt at transmission. Frames involved in any collision are not counted nor are frames that experienced a transmit underrun.

### 39.8.66 GMAC Carrier Sense Errors Register

**Name:** GMAC_CSE
**Offset:** 0x14C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CSR[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CSR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – CSR[9:0]** Carrier Sense Error
This register counts the number of frames transmitted where carrier sense was not seen during transmission or where carrier sense was deasserted after being asserted in a transmit frame without collision (no underrun). Only incremented in half duplex mode. The only effect of a carrier sense error is to increment this register. The behavior of the other statistics registers is unaffected by the detection of a carrier sense error.

### 39.8.67 GMAC Octets Received Low Register

**Name:** GMAC_ORLO
**Offset:** 0x150
**Reset:** 0x00000000
**Property:** Read-only

When reading the Octets Transmitted and Octets Received Registers, bits [31:0] should be read prior to bits [47:32] to ensure reliable operation.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RXO[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RXO[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RXO[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXO[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RXO[31:0]** Received Octets
Received octets in frame without errors [31:0]. The number of octets received in valid frames of any type. This counter is 48-bits and is read through two registers. This count does not include octets from pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 39.8.68 GMAC Octets Received High Register

**Name:** GMAC_ORHI
**Offset:** 0x154
**Reset:** 0x00000000
**Property:** Read-only

When reading the Octets Transmitted and Octets Received Registers, bits 31:0 should be read prior to bits 47:32 to ensure reliable operation.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RXO[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXO[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RXO[15:0]** Received Octets
Received octets in frame without errors [47:32]. The number of octets received in valid frames of any type. This counter is 48-bits and is read through two registers. This count does not include octets from pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 39.8.69 GMAC Frames Received Register

**Name:** GMAC_FR
**Offset:** 0x158
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – FRX[31:0]** Frames Received without Error
Frames received without error. This register counts the number of frames successfully received. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 39.8.70 GMAC Broadcast Frames Received Register

**Name:** GMAC_BCFR
**Offset:** 0x15C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – BFRX[31:0]** Broadcast Frames Received without Error
Broadcast frames received without error. This register counts the number of broadcast frames successfully received. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 39.8.71 GMAC Multicast Frames Received Register

**Name:** GMAC_MFR
**Offset:** 0x160
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | MFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – MFRX[31:0]** Multicast Frames Received without Error
This register counts the number of multicast frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 39.8.72 GMAC Pause Frames Received Register

**Name:** GMAC_PFR
**Offset:** 0x164
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – PFRX[15:0]** Pause Frames Received Register
This register counts the number of pause frames received without error.

### 39.8.73 GMAC 64 Byte Frames Received Register

**Name:** GMAC_BFR64
**Offset:** 0x168
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFRX[31:0]** 64 Byte Frames Received without Error
This register counts the number of 64 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 39.8.74 GMAC 65 to 127 Byte Frames Received Register

**Name:** GMAC_TBFR127
**Offset:** 0x16C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFRX[31:0]** 65 to 127 Byte Frames Received without Error
This register counts the number of 65 to 127 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 39.8.75 GMAC 128 to 255 Byte Frames Received Register

**Name:** GMAC_TBFR255
**Offset:** 0x170
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFRX[31:0]** 128 to 255 Byte Frames Received without Error
This register counts the number of 128 to 255 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 39.8.76 GMAC 256 to 511 Byte Frames Received Register

**Name:** GMAC_TBFR511
**Offset:** 0x174
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFRX[31:0]** 256 to 511 Byte Frames Received without Error
This register counts the number of 256 to 511 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 39.8.77 GMAC 512 to 1023 Byte Frames Received Register

**Name:** GMAC_TBFR1023
**Offset:** 0x178
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFRX[31:0]** 512 to 1023 Byte Frames Received without Error
This register counts the number of 512 to 1023 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 39.8.78 GMAC 1024 to 1518 Byte Frames Received Register

**Name:** GMAC_TBFR1518
**Offset:** 0x17C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFRX[31:0]** 1024 to 1518 Byte Frames Received without Error
This register counts the number of 1024 to 1518 byte frames successfully received without error, i.e., no underrun and not too many retries.

### 39.8.79 GMAC 1519 to Maximum Byte Frames Received Register

**Name:** GMAC_TMXBFR
**Offset:** 0x180
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFRX[31:0]** 1519 to Maximum Byte Frames Received without Error
This register counts the number of 1519 byte or above frames successfully received without error. Maximum frame size is determined by the Network Configuration Register bit 8 (1536 maximum frame size) or bit 3 (jumbo frame size). Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory. See GMAC Network Configuration Register.

### 39.8.80 GMAC Undersized Frames Received Register

**Name:** GMAC_UFR
**Offset:** 0x184
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | UFRX[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – UFRX[9:0]** Undersize Frames Received
This register counts the number of frames received less than 64 bytes in length (10/100 mode, full duplex) that do not have either a CRC error or an alignment error.

### 39.8.81 GMAC Oversized Frames Received Register

**Name:** GMAC_OFR
**Offset:** 0x188
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | OFRX[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – OFRX[9:0]** Oversized Frames Received
This register counts the number of frames received exceeding 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration Register ) in length but do not have either a CRC error, an alignment error nor a receive symbol error. See GMAC Network Configuration Register.

### 39.8.82 GMAC Jabbers Received Register

**Name:** GMAC_JR
**Offset:** 0x18C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | JRX[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | JRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – JRX[9:0]** Jabbers Received
The register counts the number of frames received exceeding 1518 bytes in length (1536 if bit 8 is set in Network Configuration Register) and have either a CRC error, an alignment error or a receive symbol error. See GMAC Network Configuration Register.

### 39.8.83 GMAC Frame Check Sequence Errors Register

**Name:** GMAC_FCSE
**Offset:** 0x190
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | FCKR[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FCKR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – FCKR[9:0]** Frame Check Sequence Errors
The register counts frames that are an integral number of bytes, have bad CRC and are between 64 and 1518 bytes in length (1536 if bit 8 is set in Network Configuration Register). This register is also incremented if a symbol error is detected and the frame is of valid length and has an integral number of bytes.

This register is incremented for a frame with bad FCS, regardless of whether it is copied to memory due to ignore FCS mode being enabled in bit 26 of the Network Configuration Register. See GMAC Network Configuration Register.

### 39.8.84 GMAC Length Field Frame Errors Register

**Name:** GMAC_LFFE
**Offset:** 0x194
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | LFER[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LFER[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – LFER[9:0]** Length Field Frame Errors
This register counts the number of frames received that have a measured length shorter than that extracted from the length field (bytes 13 and 14). This condition is only counted if the value of the length field is less than 0x0600, the frame is not of excessive length and checking is enabled through bit 16 of the Network Configuration Register. See GMAC Network Configuration Register.

### 39.8.85 GMAC Receive Symbol Errors Register

**Name:** GMAC_RSE
**Offset:** 0x198
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

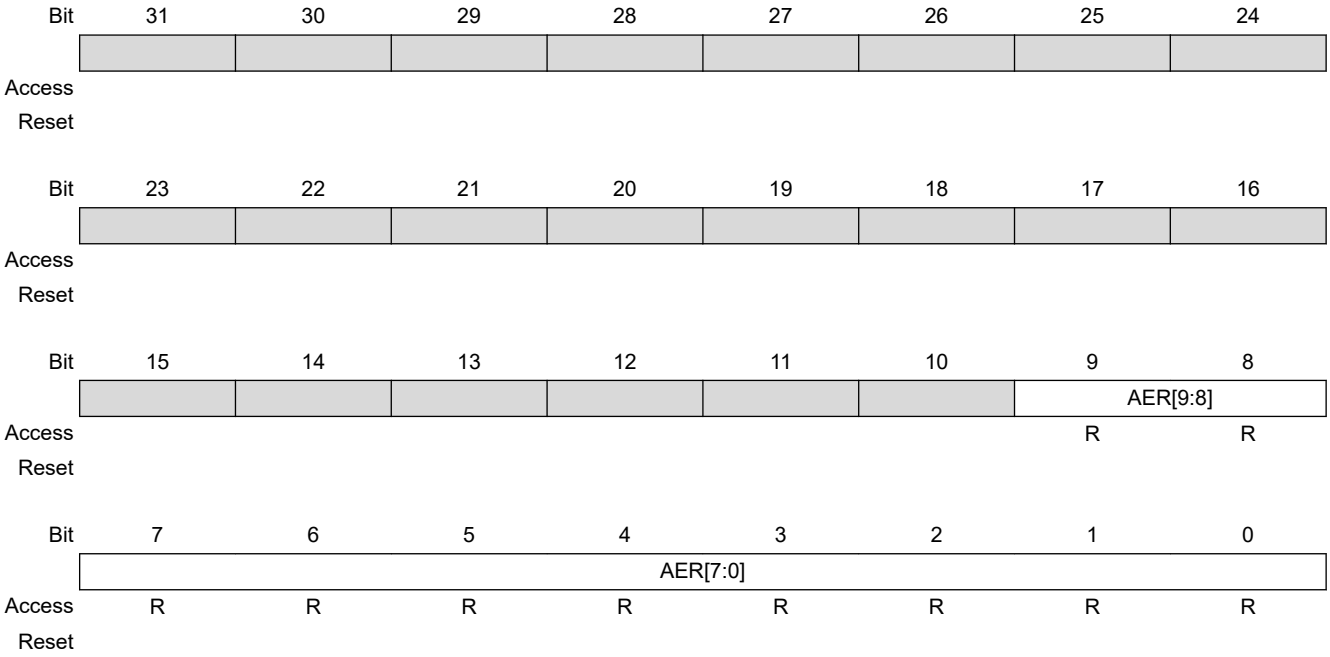
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXSE[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXSE[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – RXSE[9:0]** Receive Symbol Errors
This register counts the number of frames that had GRXER asserted during reception. For 10/100 mode symbol errors are counted regardless of frame length checks. Receive symbol errors will also be counted as an FCS or alignment error if the frame is between 64 and 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration Register). If the frame is larger it will be recorded as a jabber error. See GMAC Network Configuration Register.

### 39.8.86 GMAC Alignment Errors Register

**Name:** GMAC_AE
**Offset:** 0x19C
**Reset:** 0x0000'0000
**Property:** Read-only




**Bits 9:0 – AER[9:0]** Alignment Errors
This register counts the frames that are not an integral number of bytes long and have bad CRC when their length is truncated to an integral number of bytes and are between 64 and 1518 bytes in length (1536 if bit 8 is set in Network Configuration Register). This register is also incremented if a symbol error is detected and the frame is of valid length and does not have an integral number of bytes. See GMAC Network Configuration Register.

### 39.8.87 GMAC Receive Resource Errors Register

**Name:** GMAC_RRE
**Offset:** 0x1A0
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXRER[17:16] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RXRER[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXRER[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 17:0 – RXRER[17:0]** Receive Resource Errors
This register counts frames that are not an integral number of bytes long and have bad CRC when their length is truncated to an integral number of bytes and are between 64 and 1518 bytes in length (1536 if bit 8 is set in Network Configuration Register). This register is also incremented if a symbol error is detected and the frame is of valid length and does not have an integral number of bytes. See GMAC Network Configuration Register.

### 39.8.88 GMAC Receive Overruns Register

**Name:** GMAC_ROE
**Offset:** 0x1A4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXOVR[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXOVR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – RXOVR[9:0]** Receive Overruns
This register counts the number of frames that are address recognized but were not copied to memory due to a receive overrun.

### 39.8.89 GMAC IP Header Checksum Errors Register

**Name:** GMAC_IHCE
**Offset:** 0x1A8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | HCKER[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – HCKER[7:0]** IP Header Checksum Errors
This register counts the number of frames discarded due to an incorrect IP header checksum, but are between 64 and 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration Register) and do not have a CRC error, an alignment error, nor a symbol error.

### 39.8.90 GMAC TCP Checksum Errors Register

**Name:** GMAC_TCE
**Offset:** 0x1AC
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCKER[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – TCKER[7:0]** TCP Checksum Errors
This register counts the number of frames discarded due to an incorrect TCP checksum, but are between 64 and 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration Register) and do not have a CRC error, an alignment error, nor a symbol error.

### 39.8.91 GMAC UDP Checksum Errors Register

**Name:** GMAC_UCE
**Offset:** 0x1B0
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UCKER[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – UCKER[7:0]** UDP Checksum Errors
This register counts the number of frames discarded due to an incorrect UDP checksum, but are between 64 and 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration Register) and do not have a CRC error, an alignment error, nor a symbol error.

### 39.8.92 GMAC 1588 Timer Increment Sub-nanoseconds Register

**Name:** GMAC_TISUBN
**Offset:** 0x1BC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LSBTIR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LSBTIR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – LSBTIR[15:0]** Lower Significant Bits of Timer Increment Register
Lower significant bits of Timer Increment Register[15:0] giving a 24-bit timer_increment counter. These bits are the sub-ns value which the 1588 timer will be incremented each clock cycle. Bit n = $2^{(n-16)}$ nsec giving a resolution of approximately 15.2E$^{-15}$ sec.

### 39.8.93 GMAC 1588 Timer Seconds High Register

**Name:** GMAC_TSH
**Offset:** 0x1C0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TCS[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – TCS[15:0]** Timer Count in Seconds
This register is writable. It increments by one when the 1588 nanoseconds counter counts to one second. It may also be incremented when the Timer Adjust Register is written.

### 39.8.94 GMAC 1588 Timer Seconds Low Register

**Name:** GMAC_TSL
**Offset:** 0x1D0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TCS[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TCS[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TCS[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – TCS[31:0]** Timer Count in Seconds
This register is writable. It increments by one when the 1588 nanoseconds counter counts to one second. It may also be incremented when the Timer Adjust Register is written.

### 39.8.95 GMAC 1588 Timer Nanoseconds Register

**Name:** GMAC_TN
**Offset:** 0x1D4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | TNS[29:24] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TNS[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TNS[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TNS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:0 – TNS[29:0]** Timer Count in Nanoseconds
This register is writable. It can also be adjusted by writes to the 1588 Timer Adjust Register. It increments by the value of the 1588 Timer Increment Register each clock cycle.

### 39.8.96 GMAC 1588 Timer Adjust Register

**Name:** GMAC_TA
**Offset:** 0x1D8
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADJ | | ITDT[29:24] | | | | | |
| Access | W | | W | W | W | W | W | W |
| Reset | – | | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ITDT[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ITDT[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ITDT[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 31 – ADJ** Adjust 1588 Timer
Write as one to subtract from the 1588 timer. Write as zero to add to it.

**Bits 29:0 – ITDT[29:0]** Increment/Decrement
The number of nanoseconds to increment or decrement the 1588 Timer Nanoseconds Register. If necessary, the 1588 Seconds Register will be incremented or decremented.

### 39.8.97 GMAC 1588 Timer Increment Register

**Name:** GMAC_TI
**Offset:** 0x1DC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NIT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ACNS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CNS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – NIT[7:0]** Number of Increments
The number of increments after which the alternative increment is used.

**Bits 15:8 – ACNS[7:0]** Alternative Count Nanoseconds
Alternative count of nanoseconds by which the 1588 Timer Nanoseconds Register will be incremented each clock cycle.

**Bits 7:0 – CNS[7:0]** Count Nanoseconds
A count of nanoseconds by which the 1588 Timer Nanoseconds Register will be incremented each clock cycle.

### 39.8.98 GMAC PTP Event Frame Transmitted Seconds Low Register

**Name:** GMAC_EFTSL
**Offset:** 0x1E0
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RUD[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RUD[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RUD[31:0]** Register Update
The register is updated with the value that the 1588 Timer Seconds Register holds when the SFD of a PTP transmit primary event crosses the MII interface. An interrupt is issued when the register is updated.

### 39.8.99 GMAC PTP Event Frame Transmitted Nanoseconds Register

**Name:** GMAC_EFTN
**Offset:** 0x1E4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | RUD[29:24] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RUD[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:0 – RUD[29:0]** Register Update
The register is updated with the value that the 1588 Timer Nanoseconds Register holds when the SFD of a PTP transmit primary event crosses the MII interface. An interrupt is issued when the register is updated.

### 39.8.100 GMAC PTP Event Frame Received Seconds Low Register

**Name:** GMAC_EFRSL
**Offset:** 0x1E8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RUD[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RUD[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RUD[31:0]** Register Update
The register is updated with the value that the 1588 Timer Seconds Register holds when the SFD of a PTP receive primary event crosses the MII interface. An interrupt is issued when the register is updated.

### 39.8.101 GMAC PTP Event Frame Received Nanoseconds Register

**Name:** GMAC_EFRN
**Offset:** 0x1EC
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | RUD[29:24] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RUD[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:0 – RUD[29:0]** Register Update
The register is updated with the value that the 1588 Timer Nanoseconds Register holds when the SFD of a PTP receive primary event crosses the MII interface. An interrupt is issued when the register is updated.

### 39.8.102 GMAC PTP Peer Event Frame Transmitted Seconds Low Register

**Name:** GMAC_PEFTSL
**Offset:** 0x1F0
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RUD[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RUD[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RUD[31:0]** Register Update
The register is updated with the value that the 1588 Timer Seconds Register holds when the SFD of a PTP transmit peer event crosses the MII interface. An interrupt is issued when the register is updated.

### 39.8.103 GMAC PTP Peer Event Frame Transmitted Nanoseconds Register

**Name:** GMAC_PEFTN
**Offset:** 0x1F4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | RUD[29:24] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RUD[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:0 – RUD[29:0]** Register Update
The register is updated with the value that the 1588 Timer Nanoseconds Register holds when the SFD of a PTP transmit peer event crosses the MII interface. An interrupt is issued when the register is updated.

### 39.8.104 GMAC PTP Peer Event Frame Received Seconds Low Register

**Name:** GMAC_PEFRSL
**Offset:** 0x1F8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RUD[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RUD[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RUD[31:0]** Register Update
The register is updated with the value that the 1588 Timer Seconds Register holds when the SFD of a PTP receive primary event crosses the MII interface. An interrupt is issued when the register is updated.

### 39.8.105 GMAC PTP Peer Event Frame Received Nanoseconds Register

**Name:** GMAC_PEFRN
**Offset:** 0x1FC
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | RUD[29:24] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RUD[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:0 – RUD[29:0]** Register Update
The register is updated with the value that the 1588 Timer Nanoseconds Register holds when the SFD of a PTP receive primary event crosses the MII interface. An interrupt is issued when the register is updated.

### 39.8.106 GMAC Received LPI Transitions

**Name:** GMAC_RXLPI
**Offset:** 0x270
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – COUNT[15:0]** Count of RX LPI transitions (cleared on read)
A count of the number of times there is a transition from receiving normal idle to receiving low power idle.

### 39.8.107 GMAC Received LPI Time

**Name:** GMAC_RXLPITIME
**Offset:** 0x274
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | LPITIME[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LPITIME[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LPITIME[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – LPITIME[23:0]** Time in LPI (cleared on read)
This field increments once every 16 MCK cycles when the bit LPI Indication (bit 7) is set in the Network Status register.

### 39.8.108 GMAC Transmit LPI Transitions

**Name:** GMAC_TXLPI
**Offset:** 0x278
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – COUNT[15:0]** Count of LPI transitions (cleared on read)
A count of the number of times the bit Enable LPI Transmission (bit 19) goes from low to high in the Network Control register.

### 39.8.109 GMAC Transmit LPI Time

**Name:** GMAC_TXLPITIME
**Offset:** 0x27C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | LPITIME[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LPITIME[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LPITIME[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – LPITIME[23:0]** Time in LPI (cleared on read)
This field increments once every 16 MCK cycles when the bit Enable LPI Transmission (bit 19) is set in the Network Control register.

### 39.8.110 GMAC Interrupt Status Register Priority Queue x

**Name:** GMAC_ISRPQx
**Offset:** 0x03FC + x*0x04 [x=0..1]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | HRESP | ROVR | | |
| Access | | | | | R/W | R/W | | |
| Reset | | | | | 0 | 0 | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| Access | R/W | R/W | R/W | | | R/W | R/W | |
| Reset | 0 | 0 | 0 | | | 0 | 0 | |

**Bit 11 – HRESP** HRESP Not OK

**Bit 10 – ROVR** Receive Overrun

**Bit 7 – TCOMP** Transmit Complete

**Bit 6 – TFC** Transmit Frame Corruption Due to AHB Error
Transmit frame corruption due to AHB error—set if an error occurs whilst midway through reading transmit frame from the AHB, including HRESP errors and buffers exhausted mid frame.

**Bit 5 – RLEX** Retry Limit Exceeded or Late Collision

**Bit 2 – RXUBR** RX Used Bit Read

**Bit 1 – RCOMP** Receive Complete

### 39.8.111 GMAC Transmit Buffer Queue Base Address Register Priority Queue x

**Name:** GMAC_TBQBAPQx
**Offset:** 0x043C + x*0x04 [x=0..1]
**Reset:** 0x00000000
**Property:** Read/Write

These registers hold the start address of the transmit buffer queues (transmit buffers descriptor lists) for the additional queues and must be initialized to the address of valid descriptors, even if the priority queues are not used.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TXBQBA[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TXBQBA[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TXBQBA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXBQBA[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – TXBQBA[29:0]** Transmit Buffer Queue Base Address
Written with the address of the start of the transmit queue.

### 39.8.112 GMAC Receive Buffer Queue Base Address Register Priority Queue x

**Name:** GMAC_RBQBAPQx
**Offset:** 0x047C + x*0x04 [x=0..1]
**Reset:** 0x00000000
**Property:** Read/Write

These registers hold the start address of the receive buffer queues (receive buffers descriptor lists) for the additional queues and must be initialized to the address of valid descriptors, even if the priority queues are not used.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RXBQBA[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RXBQBA[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RXBQBA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXBQBA[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – RXBQBA[29:0]** Receive Buffer Queue Base Address
Written with the address of the start of the receive queue.

### 39.8.113 GMAC Receive Buffer Size Register Priority Queue x

**Name:** GMAC_RBSRPQx
**Offset:** 0x049C + x*0x04 [x=0..1]
**Reset:** 0x00000002
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RBS[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RBS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

**Bits 15:0 – RBS[15:0]** Receive Buffer Size
DMA receive buffer size in AHB system memory. The value defined by these bits determines the size of buffer to use in main AHB system memory when writing received data.

The value is defined in multiples of 64 bytes such that a value of 0x01 corresponds to buffers of 64 bytes, 0x02 corresponds to 128 bytes etc.

For example:

0x02: 128bytes

0x18: 1536bytes (1 × max length frame/buffer)

0xA0: 10240bytes (1 × 10K jumbo frame/buffer)

Note that this value should never be written as zero.

### 39.8.114 GMAC Credit-Based Shaping Control Register

**Name:** GMAC_CBSCR
**Offset:** 0x4BC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | QAE | QBE |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – QAE** Queue A CBS Enable

| Value | Description |
|---|---|
| 0 | Credit-based shaping on the second highest priority queue (queue A) is disabled. |
| 1 | Credit-based shaping on the second highest priority queue (queue A) is enabled. |

**Bit 0 – QBE** Queue B CBS Enable

| Value | Description |
|---|---|
| 0 | Credit-based shaping on the highest priority queue (queue B) is disabled. |
| 1 | Credit-based shaping on the highest priority queue (queue B) is enabled. |

### 39.8.115 GMAC Credit-Based Shaping IdleSlope Register for Queue A

**Name:** GMAC_CBSISQA
**Offset:** 0x4C0
**Reset:** 0x00000000
**Property:** Read/Write

Credit-based shaping must be disabled in GMAC_CBSCR before updating this register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IS[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IS[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IS[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – IS[31:0]** IdleSlope
IdleSlope value for queue A in bytes/second.

The IdleSlope value is defined as the rate of change of credit when a packet is waiting to be sent. This must not exceed the port transmit rate which is dependent on the speed of operation, e.g., 100 Mb/second = 32'h017D7840

If 50% of bandwidth was to be allocated to a particular queue in 100 Mb/second mode, then the IdleSlope value for that queue would be calculated as 32'h017D7840 / 2.

### 39.8.116 GMAC Credit-Based Shaping IdleSlope Register for Queue B

**Name:** GMAC_CBSISQB
**Offset:** 0x4C4
**Reset:** 0x00000000
**Property:** Read/Write

Credit-based shaping must be disabled in GMAC_CBSCR before updating this register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IS[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IS[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IS[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – IS[31:0]** IdleSlope
IdleSlope value for queue B in bytes/second.

The IdleSlope value is defined as the rate of change of credit when a packet is waiting to be sent. This must not exceed the port transmit rate which is dependent on the speed of operation, e.g., 100 Mb/second = 32'h017D7840.

If 50% of bandwidth was to be allocated to a particular queue in 100 Mb/second mode, then the IdleSlope value for that queue would be calculated as 32'h017D7840 / 2.

### 39.8.117 GMAC Screening Type 1 Register x Priority Queue

**Name:** GMAC_ST1RPQx
**Offset:** 0x0500 + x*0x04 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

Screening type 1 registers are used to allocate up to 3 priority queues to received frames based on certain IP or UDP fields of incoming frames.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | UDPE | DSTCE | UDPM[15:12] | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UDPM[11:4] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UDPM[3:0] | | | | DSTCM[7:4] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DSTCM[3:0] | | | | | QNB[2:0] | | |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bit 29 – UDPE** UDP Port Match Enable
When UDP port match enable is set (bit 29), the UDP Destination Port of the received UDP frame is matched against bits 27:12.

**Bit 28 – DSTCE** Differentiated Services or Traffic Class Match Enable
When DS/TC match enable is set (bit 28), the DS (differentiated services) field of the received IPv4 header or TC field (traffic class) of IPv6 headers are matched against bits 11:4.

**Bits 27:12 – UDPM[15:0]** UDP Port Match
When UDP port match enable is set (bit 29), the UDP Destination Port of the received UDP frame is matched against bits 27:12.

**Bits 11:4 – DSTCM[7:0]** Differentiated Services or Traffic Class Match
When DS/TC match enable is set (bit 28), the DS (differentiated services) field of the received IPv4 header or TC field (traffic class) of IPv6 headers are matched against bits 11:4.

**Bits 2:0 – QNB[2:0]** Queue Number (0–2)
If a match is successful, then the queue value programmed in bits 2:0 is allocated to the frame.

### 39.8.118 GMAC Screening Type 2 Register x Priority Queue

**Name:** GMAC_ST2RPQx
**Offset:** 0x0540 + x*0x04 [x=0..7]
**Reset:** 0x00000000
**Property:** Read/Write

Screening type 2 registers are used to allocate up to 3 priority queues to received frames based on the VLAN priority field of received ethernet frames.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | COMPCE | COMPC[4:0] | | | | | COMPBE |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COMPA[2:0] | | | ETHE | I2ETH[2:0] | | | VLANE |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | VLANP[2:0] | | | | QNB[2:0] | | |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bit 30 – COMPCE** Compare C Enable

| Value | Description |
|---|---|
| 0 | Comparison via the register designated by index COMPC is disabled. |
| 1 | Comparison via the register designated by index COMPC is enabled. |

**Bits 29:25 – COMPC[4:0]** Index of Screening Type 2 Compare Word 0/Word 1 register x
COMPC is a pointer to the compare registers GMAC_ST2CW0x and GMAC_ST2CW1x. When COMPCE is set, the compare is true if the data at the frame offset ANDed with the value MASKVAL is equal to the value of COMPVAL ANDed with the value of MASKVAL.

**Bit 24 – COMPBE** Compare B Enable

| Value | Description |
|---|---|
| 0 | Comparison via the register designated by index COMPB is disabled. |
| 1 | Comparison via the register designated by index COMPB is enabled. |

**Bits 23:19 – COMPB[4:0]** Index of Screening Type 2 Compare Word 0/Word 1 register x
COMPB is a pointer to the compare registers GMAC_ST2CW0x and GMAC_ST2CW1x. When COMPBE is set, the compare is true if the data at the frame offset ANDed with the value MASKVAL is equal to the value of COMPVAL ANDed with the value of MASKVAL.

**Bit 18 – COMPAE** Compare A Enable

| Value | Description |
|---|---|
| 0 | Comparison via the register designated by index COMPA is disabled. |
| 1 | Comparison via the register designated by index COMPA is enabled. |

**Bits 17:13 – COMPA[4:0]** Index of Screening Type 2 Compare Word 0/Word 1 register x
COMPA is a pointer to the compare registers GMAC_ST2CW0x and GMAC_ST2CW1x. When COMPAE is set, the compare is true if the data at the frame offset ANDed with the value MASKVAL is equal to the value of COMPVAL ANDed with the value of MASKVAL.

**Bit 12 – ETHE** EtherType Enable

| Value | Description |
|---|---|
| 0 | EtherType match with bits 15:0 in the register designated by the value of I2ETH is disabled. |
| 1 | EtherType match with bits 15:0 in the register designated by the value of I2ETH is enabled. |

**Bits 11:9 – I2ETH[2:0]** Index of Screening Type 2 EtherType register x
When ETHE is set (bit 12), the field EtherType (last EtherType in the header if the frame is VLAN tagged) is compared with bits 15:0 in the register designated by the value of I2ETH.

**Bit 8 – VLANE** VLAN Enable

| Value | Description |
|---|---|
| 0 | VLAN match is disabled. |
| 1 | VLAN match is enabled. |

**Bits 6:4 – VLANP[2:0]** VLAN Priority
When VLAN match enable is set (bit 8), the VLAN priority field of the received frame is matched against bits 7:4 of this register.

**Bits 2:0 – QNB[2:0]** Queue Number (0–2)
If a match is successful, then the queue value programmed in QNB is allocated to the frame.

### 39.8.119 GMAC Interrupt Enable Register Priority Queue x

**Name:** GMAC_IERPQx
**Offset:** 0x05FC + x*0x04 [x=0..1]
**Reset:** –
**Property:** Write-only

The following values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | HRESP | ROVR | | |
| Access | | | | | W | W | | |
| Reset | | | | | – | – | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| Access | W | W | W | | | W | W | |
| Reset | – | – | – | | | – | – | |

**Bit 11 – HRESP** HRESP Not OK

**Bit 10 – ROVR** Receive Overrun

**Bit 7 – TCOMP** Transmit Complete

**Bit 6 – TFC** Transmit Frame Corruption Due to AHB Error

**Bit 5 – RLEX** Retry Limit Exceeded or Late Collision

**Bit 2 – RXUBR** RX Used Bit Read

**Bit 1 – RCOMP** Receive Complete

### 39.8.120 GMAC Interrupt Disable Register Priority Queue x

**Name:** GMAC_IDRPQx
**Offset:** 0x061C + x*0x04 [x=0..1]
**Reset:** –
**Property:** Write-only

The following values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | HRESP | ROVR | | |
| Access | | | | | W | W | | |
| Reset | | | | | – | – | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| Access | W | W | W | | | W | W | |
| Reset | – | – | – | | | – | – | |

**Bit 11 – HRESP** HRESP Not OK

**Bit 10 – ROVR** Receive Overrun

**Bit 7 – TCOMP** Transmit Complete

**Bit 6 – TFC** Transmit Frame Corruption Due to AHB Error

**Bit 5 – RLEX** Retry Limit Exceeded or Late Collision

**Bit 2 – RXUBR** RX Used Bit Read

**Bit 1 – RCOMP** Receive Complete

### 39.8.121 GMAC Interrupt Mask Register Priority Queue x

**Name:** GMAC_IMRPQx
**Offset:** 0x063C + x*0x04 [x=0..1]
**Reset:** 0x00000000
**Property:** Read/Write

A read of this register returns the value of the receive complete interrupt mask.

A write to this register directly affects the state of the corresponding bit in the Interrupt Status Register, causing an interrupt to be generated if a 1 is written.

The following values are valid for all listed bit names of this register:

0: Corresponding interrupt is enabled.

1: Corresponding interrupt is disabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | R/W | R/W | | |
| Reset | | | | | 0 | 0 | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | R/W | R/W | R/W | | | R/W | R/W | |
| Reset | 0 | 0 | 0 | | | 0 | 0 | |

**Bit 11 – HRESP** HRESP Not OK

**Bit 10 – ROVR** Receive Overrun

**Bit 7 – TCOMP** Transmit Complete

**Bit 6 – AHB** AHB Error

**Bit 5 – RLEX** Retry Limit Exceeded or Late Collision

**Bit 2 – RXUBR** RX Used Bit Read

**Bit 1 – RCOMP** Receive Complete

### 39.8.122 GMAC Screening Type 2 EtherType Register x

**Name:** GMAC_ST2ERx
**Offset:** 0x06E0 + x*0x04 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COMPVAL[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COMPVAL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – COMPVAL[15:0]** Ethertype Compare Value
When the bit GMAC_ST2RPQ.ETHE is enabled, the EtherType (last EtherType in the header if the frame is VLAN tagged) is compared with bits 15:0 in the register designated by GMAC_ST2RPQ.I2ETH.

### 39.8.123 GMAC Screening Type 2 Compare Word 0 Register x

**Name:** GMAC_ST2CW0x
**Offset:** 0x0700 + x*0x08 [x=0..23]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | COMPVAL[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | COMPVAL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MASKVAL[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MASKVAL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – COMPVAL[15:0]** Compare Value
The byte stored in bits [23:16] is compared against the first byte of the 2 bytes extracted from the frame.

The byte stored in bits [31:24] is compared against the second byte of the 2 bytes extracted from the frame.

**Bits 15:0 – MASKVAL[15:0]** Mask Value
The value of MASKVAL ANDed with the 2 bytes extracted from the frame is compared to the value of MASKVAL ANDed with the value of COMPVAL.

### 39.8.124 GMAC Screening Type 2 Compare Word 1 Register x

**Name:** GMAC_ST2CW1x
**Offset:** 0x0704 + x*0x08 [x=0..23]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | OFFSSTRT[1:1] |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OFFSSTRT[0:0] | OFFSVAL[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 8:7 – OFFSSTRT[1:0]** Ethernet Frame Offset Start

| Value | Name | Description |
|---|---|---|
| 0 | FRAMESTART | Offset from the start of the frame |
| 1 | ETHERTYPE | Offset from the byte after the EtherType field |
| 2 | IP | Offset from the byte after the IP header field |
| 3 | TCP_UDP | Offset from the byte after the TCP/UDP header field |

**Bits 6:0 – OFFSVAL[6:0]** Offset Value in Bytes
The value of OFFSVAL ranges from 0 to 127 bytes, and is counted from either the start of the frame, the byte after the EtherType field (last EtherType in the header if the frame is VLAN tagged), the byte after the IP header (IPv4 or IPv6) or the byte after the TCP/UDP header.

# 40. USB High Speed Device Port (UDPHS)

## 40.1 Description

The USB High Speed Device Port (UDPHS) is compliant with the Universal Serial Bus (USB), rev 2.0 High Speed device specification.

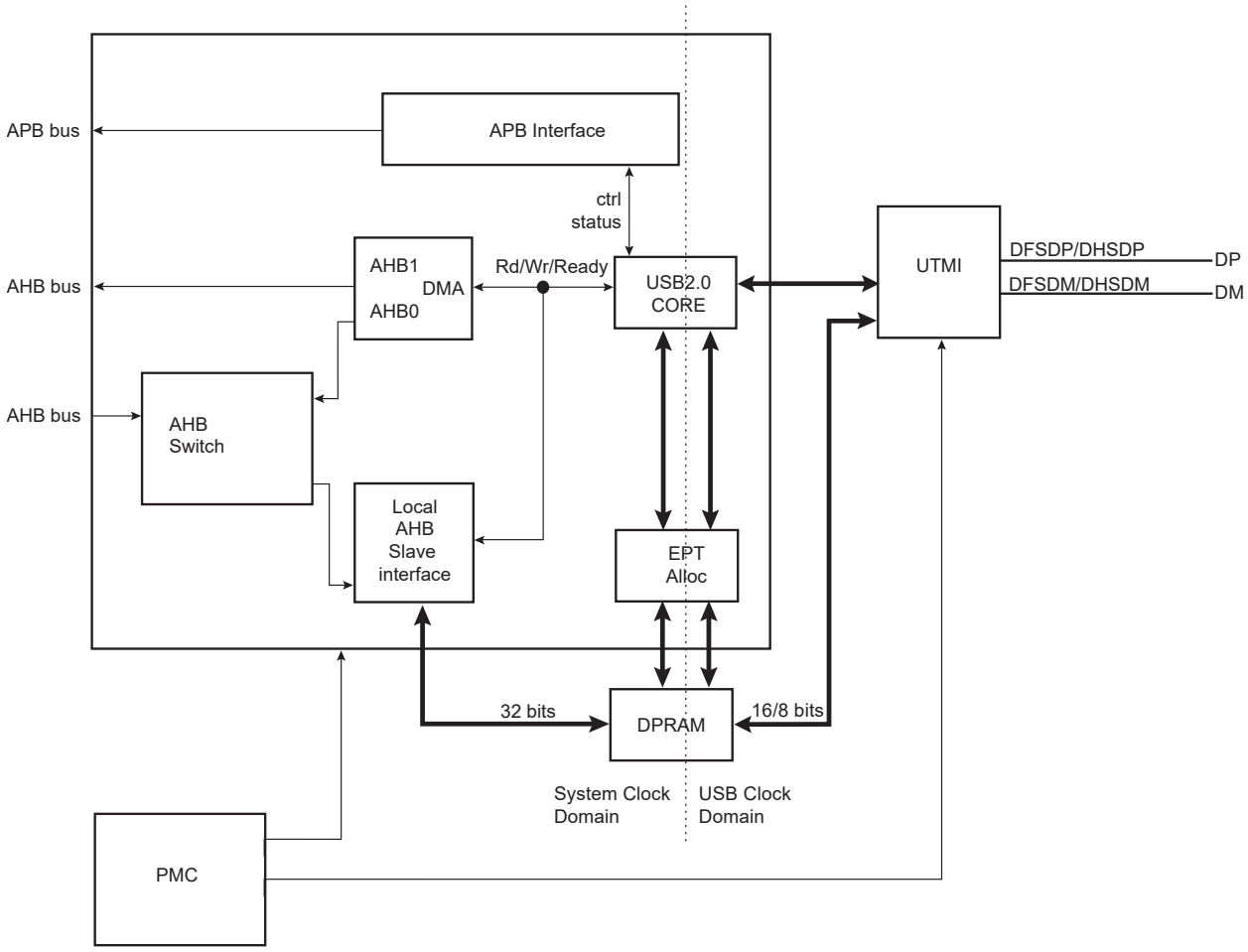
Each endpoint can be configured in one of several USB transfer types. It can be associated with one, two or three banks of a Dual-port RAM used to store the current data payload. If two or three banks are used, one DPR bank is read or written by the processor, while the other is read or written by the USB device peripheral. This feature is mandatory for isochronous endpoints.

## 40.2 Embedded Characteristics

- 1 High-speed Device
- 1 UTMI transceiver shared between Host and Device
- USB v2.0 High Speed (480 Mbits/s) Compliant
- 16 Endpoints up to 1024 bytes
- Embedded Dual-port RAM for Endpoints
- Suspend/Resume Logic (Command of UTMI)
- Up to Three Memory Banks for Endpoints (Not for Control Endpoint)
- 8 Kbytes of DPRAM

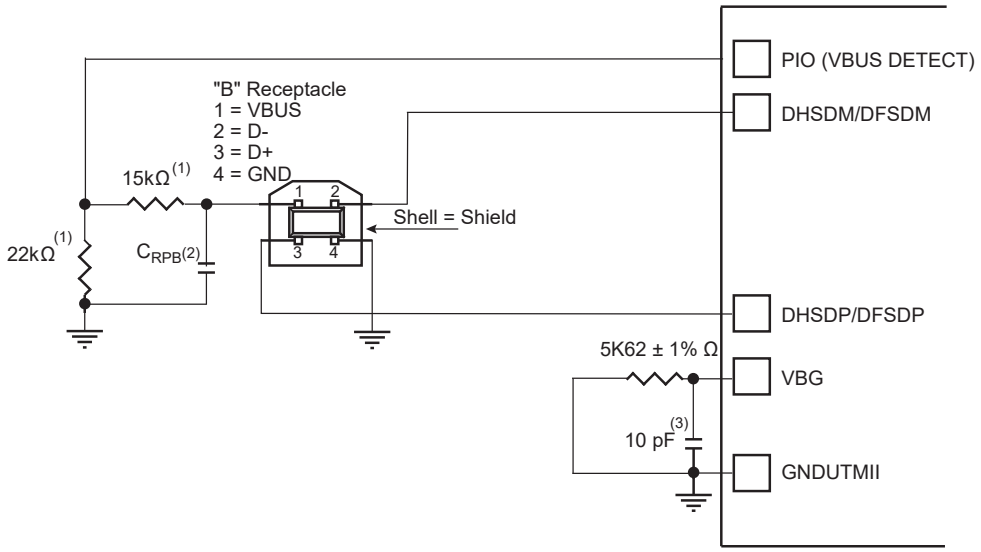
## 40.3 Block Diagram

**Figure 40-1. Block Diagram**



## 40.4 Typical Connection

**Figure 40-2. Board Schematic**




**Note:**

1. The values shown on the 22 kΩ and 15 kΩ resistors are only valid with 3V3-supplied PIOs.
2. CRPB: Upstream Facing Port Bypass Capacitance of 1 µF to 10 µF (refer to “DC Electrical Characteristics” in Universal Serial Bus Specification Rev. 2)
3. 10 pF capacitor on VBG is a provision and may not be populated.

## 40.5 Product Dependencies

### 40.5.1 Power Management

The UDPHS is not continuously clocked.

To use the UDPHS, the programmer must first enable the UDPHS Clock in the Power Management Controller Peripheral Clock Enable Register (PMC_PCER), then enable the PLL in the PMC UTMI Clock Configuration Register (CKGR_UCKR). Finally, the programmer must enable BIAS in CKGR_UCKR.

However, if the application does not require UDPHS operations, the UDPHS clock can be stopped when not needed and restarted later.

### 40.5.2 Interrupt Sources

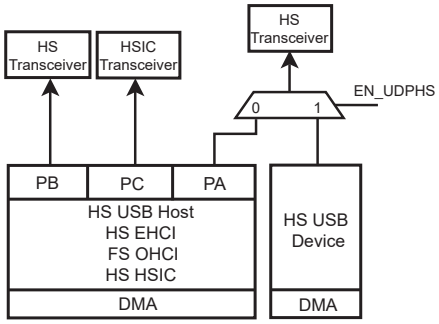
The UDPHS interrupt line is connected on one of the internal sources of the interrupt controller. Using the UDPHS interrupt requires the interrupt controller to be programmed first.

## 40.6 Functional Description

### 40.6.1 UTMI Transceivers Sharing

The High Speed USB Host Port A is shared with the High Speed USB Device port and connected to the second UTMI transceiver. The selection between Host Port A and USB Device is controlled by the UDPHS enable bit (EN_UDPHS) located in the UDPHS_CTRL register.

**Figure 40-3. USB Selection**




### 40.6.2 USB V2.0 High Speed Device Port Introduction

The USB V2.0 High Speed Device Port provides communication services between host and attached USB devices. Each device is offered with a collection of communication flows (pipes) associated with each endpoint. Software on the host communicates with a USB Device through a set of communication flows.

### 40.6.3 USB V2.0 High Speed Transfer Types

A communication flow is carried over one of four transfer types defined by the USB device.

A device provides several logical communication pipes with the host. To each logical pipe is associated an endpoint. Transfer through a pipe belongs to one of the four transfer types:

- Control Transfers: Used to configure a device at attach time and can be used for other device-specific purposes, including control of other pipes on the device.
- Bulk Data Transfers: Generated or consumed in relatively large burst quantities and have wide dynamic latitude in transmission constraints.
- Interrupt Data Transfers: Used for timely but reliable delivery of data, for example, characters or coordinates with human-perceptible echo or feedback response characteristics.
- Isochronous Data Transfers: Occupy a prenegotiated amount of USB bandwidth with a prenegotiated delivery latency. (Also called streaming real time transfers.)

As indicated below, transfers are sequential events carried out on the USB bus.

Endpoints must be configured according to the transfer type they handle.

**Table 40-1. USB Communication Flow**

| Transfer | Direction | Bandwidth | Endpoint Size | Error Detection | Retrying |
|---|---|---|---|---|---|
| Control | Bidirectional | Not guaranteed | 8, 16, 32, 64 | Yes | Automatic |
| Isochronous | Unidirectional | Guaranteed | 8–1024 | Yes | No |
| Interrupt | Unidirectional | Not guaranteed | 8–1024 | Yes | Yes |
| Bulk | Unidirectional | Not guaranteed | 8–512 | Yes | Yes |

### 40.6.4 USB Transfer Event Definitions

A transfer is composed of one or several transactions as shown in the table below.

**Table 40-2. USB Transfer Events**

| Transfer | | Transaction |
|---|---|---|
| Direction | Type | |
| CONTROL (bidirectional) | Control Transfer (1) | • Setup transaction → Data IN transactions → Status OUT transaction<br>• Setup transaction → Data OUT transactions → Status IN transaction<br>• Setup transaction → Status IN transaction |
| IN (device toward host) | Bulk IN Transfer | • Data IN transaction → Data IN transaction |
| | Interrupt IN Transfer | • Data IN transaction → Data IN transaction |
| | Isochronous IN Transfer (2) | • Data IN transaction → Data IN transaction |
| OUT (host toward device) | Bulk OUT Transfer | • Data OUT transaction → Data OUT transaction |
| | Interrupt OUT Transfer | • Data OUT transaction → Data OUT transaction |
| | Isochronous OUT Transfer (2) | • Data OUT transaction → Data OUT transaction |

**Note:**

1. Control transfer must use endpoints with one bank and can be aborted using a stall handshake.
2. Isochronous transfers must use endpoints configured with two or three banks.

An endpoint handles all transactions related to the type of transfer for which it has been configured.

**Table 40-3. UDPHS Endpoint Description**

| Endpoint # | Mnemonic | Nb Bank | DMA | High Band Width | Max. Endpoint Size | Endpoint Type |
|---|---|---|---|---|---|---|
| 0 | EPT_0 | 1 | N | N | 64 | Control |
| 1 | EPT_1 | 3 | Y | Y | 1024 | Ctrl/Bulk/Iso(1)/Interrupt |
| 2 | EPT_2 | 3 | Y | Y | 1024 | Ctrl/Bulk/Iso(1)/Interrupt |
| 3 | EPT_3 | 2 | Y | N | 1024 | Ctrl/Bulk/Iso(1)/Interrupt |
| 4 | EPT_4 | 2 | Y | N | 1024 | Ctrl/Bulk/Iso(1)/Interrupt |
| 5 | EPT_5 | 2 | Y | N | 1024 | Ctrl/Bulk/Iso(1)/Interrupt |
| 6 | EPT_6 | 2 | Y | N | 1024 | Ctrl/Bulk/Iso(1)/Interrupt |
| 7 | EPT_7 | 2 | Y | N | 1024 | Ctrl/Bulk/Iso(1)/Interrupt |
| 8 | EPT_8 | 2 | N | N | 1024 | Ctrl/Bulk/Iso(1)/Interrupt |
| 9 | EPT_9 | 2 | N | N | 1024 | Ctrl/Bulk/Iso(1)/Interrupt |
| 10 | EPT_10 | 2 | N | N | 1024 | Ctrl/Bulk/Iso(1)/Interrupt |
| 11 | EPT_11 | 2 | N | N | 1024 | Ctrl/Bulk/Iso(1)/Interrupt |

| .....continued | | | | | | |
|---|---|---|---|---|---|---|
| Endpoint # | Mnemonic | Nb Bank | DMA | High Band Width | Max. Endpoint Size | Endpoint Type |
| 12 | EPT_12 | 2 | N | N | 1024 | Ctrl/Bulk/Iso(1)/Interrupt |
| 13 | EPT_13 | 2 | N | N | 1024 | Ctrl/Bulk/Iso(1)/Interrupt |
| 14 | EPT_14 | 2 | N | N | 1024 | Ctrl/Bulk/Iso(1)/Interrupt |
| 15 | EPT_15 | 2 | N | N | 1024 | Ctrl/Bulk/Iso(1)/Interrupt |

**Note:**

1. In Isochronous (Iso) mode, it is preferable that High Band Width capability is available.

The size of the internal DPRAM is 8 Kbytes.

Suspend and resume are automatically detected by the UDPHS device, which notifies the processor by raising an interrupt.

### 40.6.5 USB V2.0 High Speed BUS Transactions

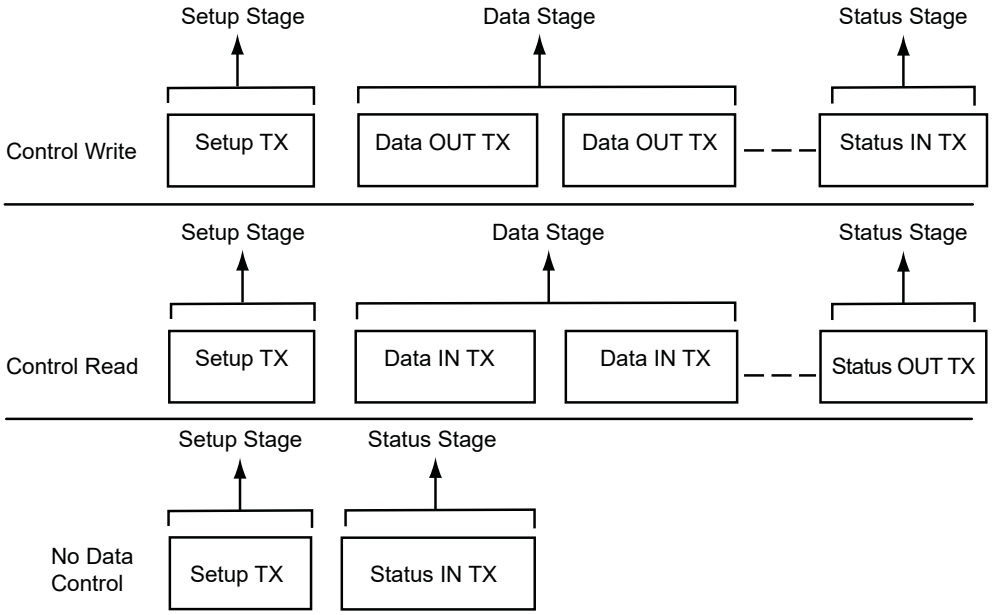
Each transfer results in one or more transactions over the USB bus.

There are five kinds of transactions flowing across the bus in packets:

1. Setup Transaction
2. Data IN Transaction
3. Data OUT Transaction
4. Status IN Transaction
5. Status OUT Transaction

A status IN or OUT transaction is identical to a data IN or OUT transaction.

**Figure 40-4. Control Read and Write Sequences**



### 40.6.6 Endpoint Configuration

The endpoint 0 is always a control endpoint, it must be programmed and active in order to be enabled when the End Of Reset interrupt occurs.

To configure the endpoints:

- Fill the configuration register (UDPHS_EPTCFG) with the endpoint size, direction (IN or OUT), type (CTRL, Bulk, IT, ISO) and the number of banks.
- Fill the number of transactions (NB_TRANS) for isochronous endpoints.

Note: For control endpoints the direction has no effect.

- Verify that the EPT_MAPD flag is set. This flag is set if the endpoint size and the number of banks are correct for this endpoint and compared to the FIFO maximum capacity and the maximum number of allowed banks.
- Configure control flags of the endpoint and enable it in UDPHS_EPTCTLENBx according to the section UDPHS Endpoint Control Disable Register (Isochronous Endpoint).

Control endpoints can generate interrupts and use only 1 bank.

All endpoints (except endpoint 0) can be configured either as Bulk, Interrupt or Isochronous. Refer to the table UDPHS Endpoint Description.

The maximum packet size they can accept corresponds to the maximum endpoint size.

**Note:** The endpoint size of 1024 is reserved for isochronous endpoints.

The size of the DPRAM is 8 Kbytes. The DPR is shared by all active endpoints. The memory size required by the active endpoints must not exceed the size of the DPRAM.

SIZE_DPRAM = SIZE _EPT0

+ NB_BANK_EPT1 x SIZE_EPT1

+ NB_BANK_EPT2 x SIZE_EPT2

+ NB_BANK_EPT3 x SIZE_EPT3

+ NB_BANK_EPT4 x SIZE_EPT4

+ NB_BANK_EPT5 x SIZE_EPT5

+ NB_BANK_EPT6 x SIZE_EPT6

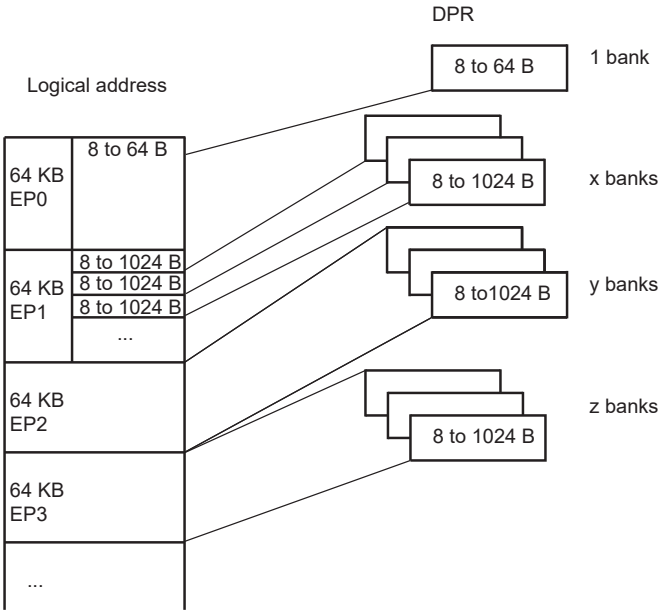
+... (refer to UDPHS Endpoint Configuration Register)

If a user tries to configure endpoints with a size the sum of which is greater than the DPRAM, then the EPT_MAPD is not set.

The application has access to the physical block of DPR reserved for the endpoint through a 64-Kbyte logical address space.

The physical block of DPR reserved for the endpoint is remapped all along the 64-Kbyte logical address space. The application can write a 64-Kbyte buffer linearly.

**Figure 40-5. Logical Address Space for DPR Access**




Configuration examples of UDPHS_EPTCTLx (UDPHS Endpoint Control Disable Register (Isochronous Endpoint)) for Bulk IN endpoint type follow below.

- With DMA
  - AUTO_VALID: Automatically validate the packet and switch to the next bank.
  - EPT_ENABL: Enable endpoint.
- Without DMA:
  - TXRDY: An interrupt is generated after each transmission.
  - EPT_ENABL: Enable endpoint.

Configuration examples of Bulk OUT endpoint type follow below.

- With DMA
  - AUTO_VALID: Automatically validate the packet and switch to the next bank.
  - EPT_ENABL: Enable endpoint.
- Without DMA
  - RXRDY_TXKL: An interrupt is sent after a new packet has been stored in the endpoint FIFO.
  - EPT_ENABL: Enable endpoint.

### 40.6.7 DPRAM Management

Endpoints can only be allocated in ascending order, from the endpoint 0 to the last endpoint to be allocated. The user shall therefore configure them in the same order.

The allocation of an endpoint x starts when the Number of Banks field in the UDPHS Endpoint Configuration Register (UDPHS_EPTCFGx.BK_NUMBER) is different from zero. Then, the hardware allocates a memory area in the DPRAM and inserts it between the x - 1 and x+ 1 endpoints. The x+ 1 endpoint memory window slides up and its data is lost. Note that the following endpoint memory windows (from x+ 2) do not slide.

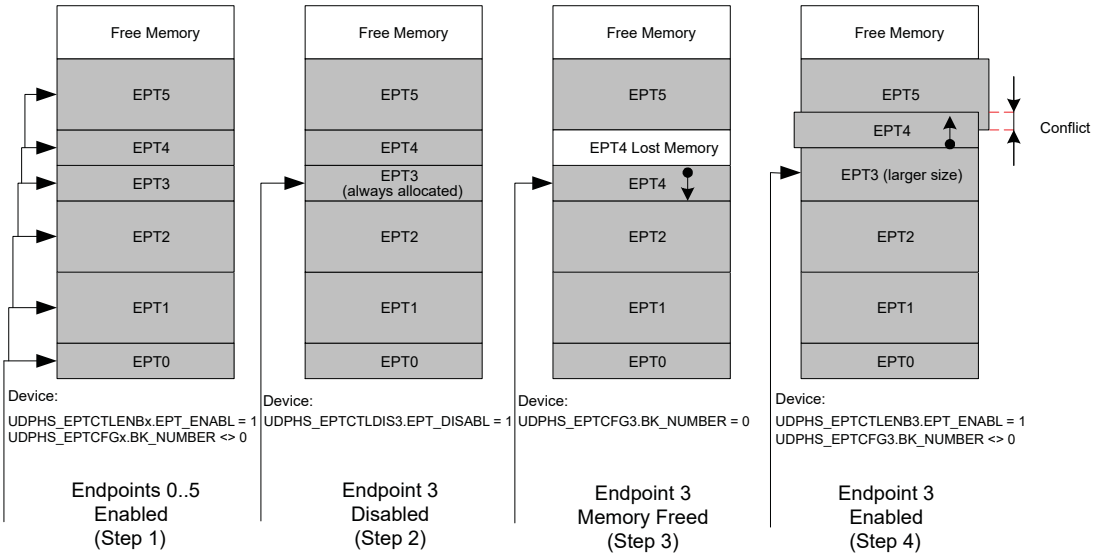
Disabling an endpoint, by writing a one to the Endpoint Disable bit in the UDPHS Endpoint Control Disable Register (UDPHS_EPTCTLDISx.EPT_DISABL), does not reset its configuration:

- Endpoint Banks (UDPHS_EPTCFGx.BK_NUMBER)
- Endpoint Size (UDPHS_EPTCFGx.EPT_SIZE)
- Endpoint Direction (UDPHS_EPTCFGx.EPT_DIR)
- Endpoint Type (UDPHS_EPTCFGx.EPT_TYPE)

To free its memory, the user shall write a zero to the UDPHS_EPTCFGx.BK_NUMBER field. The x+ 1 endpoint memory window then slides down and its data is lost. Note that the following endpoint memory windows (from x + 2) do not slide.

The following figure illustrates the allocation and reorganization of the DPRAM in a typical example.

**Figure 40-6. Example of DPRAM Allocation and Reorganization**




DPRAM allocation sequence:

1. The endpoints 0 to 5 are enabled, configured and allocated in ascending order. Each endpoint then owns a memory area in the DPRAM.
2. The endpoint 3 is disabled, but its memory is kept allocated by the controller.
3. In order to free its memory, its UDPHS_EPTCFGx.BK_NUMBER field is written to zero. The endpoint 4 memory window slides down, but the endpoint 5 does not move.
4. If the user chooses to reconfigure the endpoint 3 with a larger size, the controller allocates a memory area after the endpoint 2 memory area and automatically slides up the endpoint 4 memory window. The endpoint 5 does not move and a memory conflict appears as the memory windows of the endpoints 4 and 5 overlap. The data of these endpoints is potentially lost.

**Note:**

1. There is no way the data of the endpoint 0 can be lost (except if it is de-allocated) as the memory allocation and de-allocation may affect only higher endpoints.
2. Deactivating then reactivating the same endpoint with the same configuration only modifies temporarily the controller DPRAM pointer and size for this endpoint. Nothing changes in the DPRAM, higher endpoints seem not to have been moved and their data is preserved as far as nothing has been written or received into them while changing the allocation state of the first endpoint.
3. When the user writes a value different from zero to the UDPHS_EPTCFGx.BK_NUMBER field, the Endpoint Mapped bit (UDPHS_EPTCFGx.EPT_MAPD) is set only if the configured size and

number of banks are correct as compared to the endpoint maximal allowed values and to the maximal FIFO size (i.e., the DPRAM size). The UDPHS_EPTCFGx.EPT_MAPD value does not consider memory allocation conflicts.

### 40.6.8 Transfer With DMA

USB packets of any length may be transferred when required by the UDPHS device. These transfers always feature sequential addressing.

Packet data AHB bursts may be locked on a DMA buffer basis for drastic overall AHB bus bandwidth performance boost with paged memories. These clock-cycle consuming memory row (or bank) changes will then likely not occur, or occur only once instead of several times, during a single big USB packet DMA transfer in case another AHB master addresses the memory. The locked bursts result in up to 128-word single-cycle unbroken AHB bursts for bulk endpoints and 256-word single-cycle unbroken bursts for isochronous endpoints.

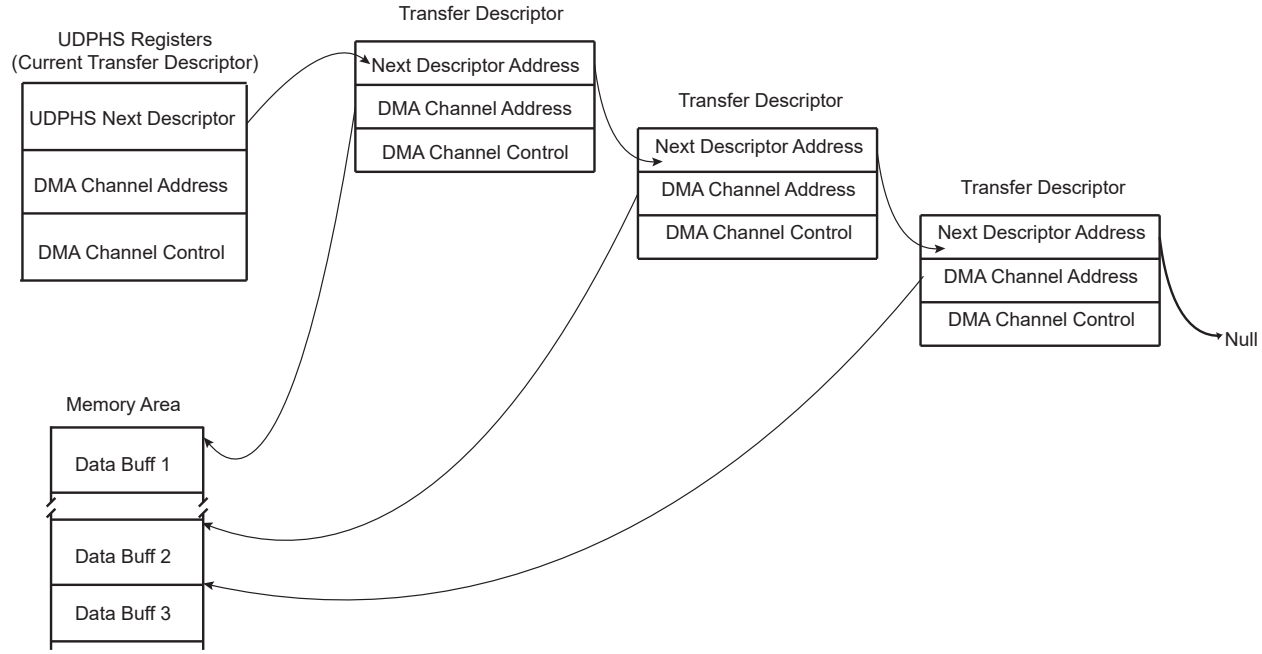
This maximum burst length is then controlled by the lowest programmed USB endpoint size (EPT_SIZE field in the UDPHS_EPTCFGx register) and DMA Size (BUFF_LENGTH field in the UDPHS_DMACONTROLx register).

The USB 2.0 device average throughput may be up to nearly 60 Mbyte/s. Its internal slave average access latency decreases as burst length increases due to the 0 wait-state side effect of unchanged endpoints. If at least 0 wait-state word burst capability is also provided by the external DMA AHB bus slaves, each of both DMA AHB busses need less than 50% bandwidth allocation for full USB 2.0 bandwidth usage at 30 MHz, and less than 25% at 60 MHz.

The UDPHS DMA Channel Transfer Descriptor is described in the section UDPHS DMA Channel Transfer Descriptor.

**Note:** In case of debug, be careful to address the DMA to an SRAM address even if a remap is done.

**Figure 40-7. Example of DMA Chained List**



### 40.6.9 Transfer Without DMA

**Important:** If the DMA is not to be used, it is necessary to disable it, otherwise it can be enabled by previous versions of software without warning. If this should occur, the DMA can process data before an interrupt without knowledge of the user.

The recommended means to disable DMA are as follows:

```
// Reset IP UDPHS
  AT91C_BASE_UDPHS->UDPHS_CTRL &= ~AT91C_UDPHS_EN_UDPHS;
  AT91C_BASE_UDPHS->UDPHS_CTRL |= AT91C_UDPHS_EN_UDPHS;//
With OR without DMA !!!
  for( i=1; i<=((AT91C_BASE_UDPHS->UDPHS_IPFEATURES &
AT91C_UDPHS_DMA_CHANNEL_NBR)>>4); i++ ) {
// RESET endpoint canal DMA:
    // DMA stop channel command
   AT91C_BASE_UDPHS->UDPHS_DMA[i].UDPHS_DMACONTROL = 0; // STOP
command
// Disable endpoint
  AT91C_BASE_UDPHS->UDPHS_EPT[i].UDPHS_EPTCTLDIS |= 0XFFFFFFFF;
// Reset endpoint config
  AT91C_BASE_UDPHS->UDPHS_EPT[i].UDPHS_EPTCTLCFG = 0;
// Reset DMA channel (Buff count and Control field)
  AT91C_BASE_UDPHS->UDPHS_DMA[i].UDPHS_DMACONTROL = 0x02; // NON
STOP command
// Reset DMA channel 0 (STOP)
  AT91C_BASE_UDPHS->UDPHS_DMA[i].UDPHS_DMACONTROL = 0; // STOP
command
// Clear DMA channel status (read the register for clear it)
  AT91C_BASE_UDPHS->UDPHS_DMA[i].UDPHS_DMASTATUS =
  AT91C_BASE_UDPHS->UDPHS_DMA[i].UDPHS_DMASTATUS;
}
```

### 40.6.10 Handling Transactions with USB V2.0 Device Peripheral

#### 40.6.10.1 Setup Transaction

The setup packet is valid in the DPR while RX_SETUP is set. Once RX_SETUP is cleared by the application, the UDPHS accepts the next packets sent over the device endpoint.

When a valid setup packet is accepted by the UDPHS:

- The UDPHS device automatically acknowledges the setup packet (sends an ACK response)
- Payload data is written in the endpoint
- Sets the RX_SETUP interrupt
- The BYTE_COUNT field in the UDPHS_EPTSTAx register is updated

An endpoint interrupt is generated while RX_SETUP in the UDPHS_EPTSTAx register is not cleared. This interrupt is carried out to the microcontroller if interrupts are enabled for this endpoint.

Thus, firmware must detect RX_SETUP polling UDPHS_EPTSTAx or catching an interrupt, read the setup packet in the FIFO, then clear the RX_SETUP bit in the UDPHS_EPTCLRSTA register to acknowledge the setup stage.

If STALL_SNT was set to 1, then this bit is automatically reset when a setup token is detected by the device. Then, the device still accepts the setup stage. (See the section STALL).

#### 40.6.10.2 NYET

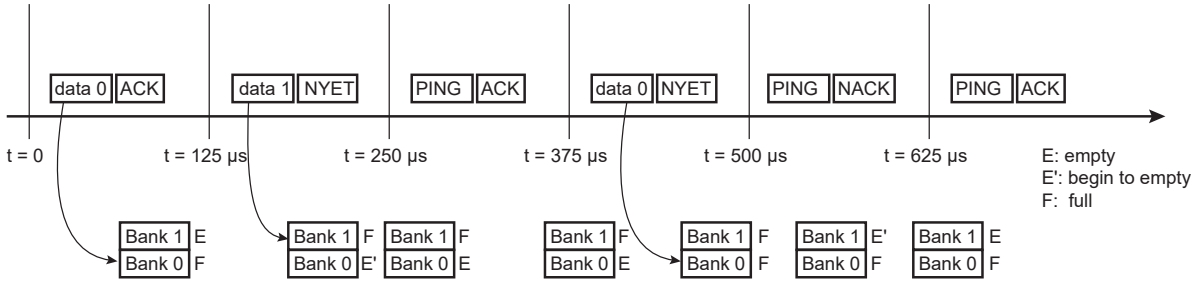
NYET is a High Speed only handshake. It is returned by a High Speed endpoint as part of the PING protocol.

High Speed devices must support an improved NAK mechanism for Bulk OUT and control endpoints (except setup stage). This mechanism allows the device to tell the host whether it has sufficient endpoint space for the next OUT transfer (refer to USB 2.0 spec 8.5.1 NAK Limiting via Ping Flow Control).

The NYET/ACK response to a High Speed Bulk OUT transfer and the PING response are automatically handled by hardware in the UDPHS_EPTCTLx register (except when the user wants to force a NAK response by using the NYET_DIS bit).

If the endpoint responds instead to the OUT/DATA transaction with an NYET handshake, this means that the endpoint accepted the data but does not have room for another data payload. The host controller must return to using a PING token until the endpoint indicates it has space available.

**Figure 40-8. NYET Example with Two Endpoint Banks**




### 40.6.10.3 Data IN

• Bulk IN or Interrupt IN

Data IN packets are sent by the device during the data or the status stage of a control transfer or during an (interrupt/bulk/isochronous) IN transfer. Data buffers are sent packet by packet under the control of the application or under the control of the DMA channel.

There are three ways for an application to transfer a buffer in several packets over the USB:

- packet by packet (see Bulk IN or Interrupt IN: Sending a Packet Under Application Control (Device to Host) below)
- 64 Kbytes (see Bulk IN or Interrupt IN: Sending a Packet Under Application Control (Device to Host) below)
- DMA (see Bulk IN or Interrupt IN: Sending a Buffer Using DMA (Device to Host) below)
  • Bulk IN or Interrupt IN: Sending a Packet Under Application Control (Device to Host)

• Bulk IN or Interrupt IN: Sending a Packet Under Application Control (Device to Host)

The application can write one or several banks.

A simple algorithm can be used by the application to send packets regardless of the number of banks associated to the endpoint.

**Algorithm Description for Each Packet**

- The application waits for the TXRDY flag to be cleared in the UDPHS_EPTSTAx register before it can perform a write access to the DPR.
- The application writes one USB packet of data in the DPR through the 64 Kbytes endpoint logical memory window.
- The application sets TXRDY flag in the UDPHS_EPTSETSTAx register.

The application is notified that it is possible to write a new packet to the DPR by the TXRDY interrupt. This interrupt can be enabled or masked by setting the TXRDY bit in the UDPHS_EPTCTLENB/ UDPHS_EPTCTLDIS register.

**Algorithm Description to Fill Several Packets**

Using the previous algorithm, the application is interrupted for each packet. It is possible to reduce the application overhead by writing linearly several banks at the same time. The AUTO_VALID bit in the UDPHS_EPTCTLx must be set by writing the AUTO_VALID bit in the UDPHS_EPTCTLENBx register.

The auto-valid-bank mechanism allows the transfer of data (IN and OUT) without the intervention of the CPU. This means that bank validation (set TXRDY or clear the RXRDY_TXKL bit) is done by hardware.

- The application checks the BUSY_BANK_STA field in the UDPHS_EPTSTAx register. The application must wait that at least one bank is free.
- The application writes a number of bytes inferior to the number of free DPR banks for the endpoint. Each time the application writes the last byte of a bank, the TXRDY signal is automatically set by the UDPHS.
- If the last packet is incomplete (i.e., the last byte of the bank has not been written) the application must set the TXRDY bit in the UDPHS_EPTSETSTAx register.

The application is notified that all banks are free, so that it is possible to write another burst of packets by the BUSY_BANK interrupt. This interrupt can be enabled or masked by setting the BUSY_BANK flag in the UDPHS_EPTCTLENB and UDPHS_EPTCTLDIS registers.

This algorithm must not be used for isochronous transfer. In this case, the ping-pong mechanism does not operate.

A Zero Length Packet can be sent by setting just the TXRDY flag in the UDPHS_EPTSETSTAx register.

• Bulk IN or Interrupt IN: Sending a Buffer Using DMA (Device to Host)

The UDPHS integrates a DMA host controller. This DMA controller can be used to transfer a buffer from the memory to the DPR or from the DPR to the processor memory under the UDPHS control. The DMA can be used for all transfer types except control transfer.

Example DMA configuration:

1. Program UDPHS_DMAADDRESS x with the address of the buffer that should be transferred.
2. Enable the interrupt of the DMA in UDPHS_IEN
3. Program UDPHS_ DMACONTROLx:
   - Size of buffer to send: size of the buffer to be sent to the host.
   - END_B_EN: The endpoint can validate the packet (according to the values programmed in the AUTO_VALID and SHRT_PCKT fields of UDPHS_EPTCTLx.) (see section UDPHS Endpoint Control Disable Register (Isochronous Endpoint) and Figure 40-13)
   - END_BUFFIT: generate an interrupt when the BUFF_COUNT in UDPHS_DMASTATUSx reaches 0.
   - CHANN_ENB: Run and stop at end of buffer

The auto-valid-bank mechanism allows the transfer of data (IN & OUT) without the intervention of the CPU. This means that bank validation (set TXRDY or clear the RXRDY_TXKL bit) is done by hardware.

A transfer descriptor can be used. Instead of programming the register directly, a descriptor should be programmed and the address of this descriptor is then given to UDPHS_DMANXTDSC to be processed after setting the LDNXT_DSC field (Load Next Descriptor Now) in UDPHS_DMACONTROLx register.

The structure that defines this transfer descriptor must be aligned.

Each buffer to be transferred must be described by a DMA Transfer descriptor (see section UDPHS DMA Channel Transfer Descriptor). Transfer descriptors are chained. Before executing transfer of the buffer, the UDPHS may fetch a new transfer descriptor from the memory address pointed by the UDPHS_DMANXTDSCx register. Once the transfer is complete, the transfer status is updated in the UDPHS_DMASTATUSx register.

To chain a new transfer descriptor with the current DMA transfer, the DMA channel must be stopped. To do so, INTDIS_DMA and TXRDY may be set in the UDPHS_EPTCTLENBx register. It is also possible for the application to wait for the completion of all transfers. In this case the LDNXT_DSC bit in the last transfer descriptor UDPHS_DMACONTROLx register must be set to 0 and the CHANN_ENB bit set to 1.

Then the application can chain a new transfer descriptor.

The INTDIS_DMA can be used to stop the current DMA transfer if an enabled interrupt is triggered. This can be used to stop DMA transfers in case of errors.

The application can be notified at the end of any buffer transfer (ENB_BUFFIT bit in the UDPHS_DMACONTROLx register).

**Figure 40-9. Data IN Transfer for Endpoint with One Bank**

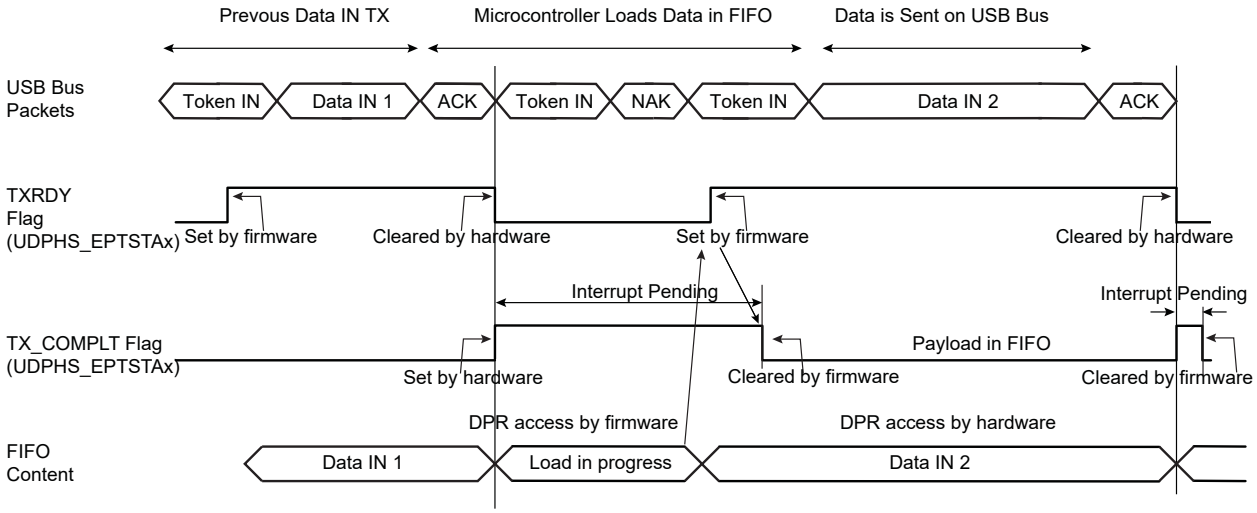


**Figure 40-10. Data IN Transfer for Endpoint with Two Banks**

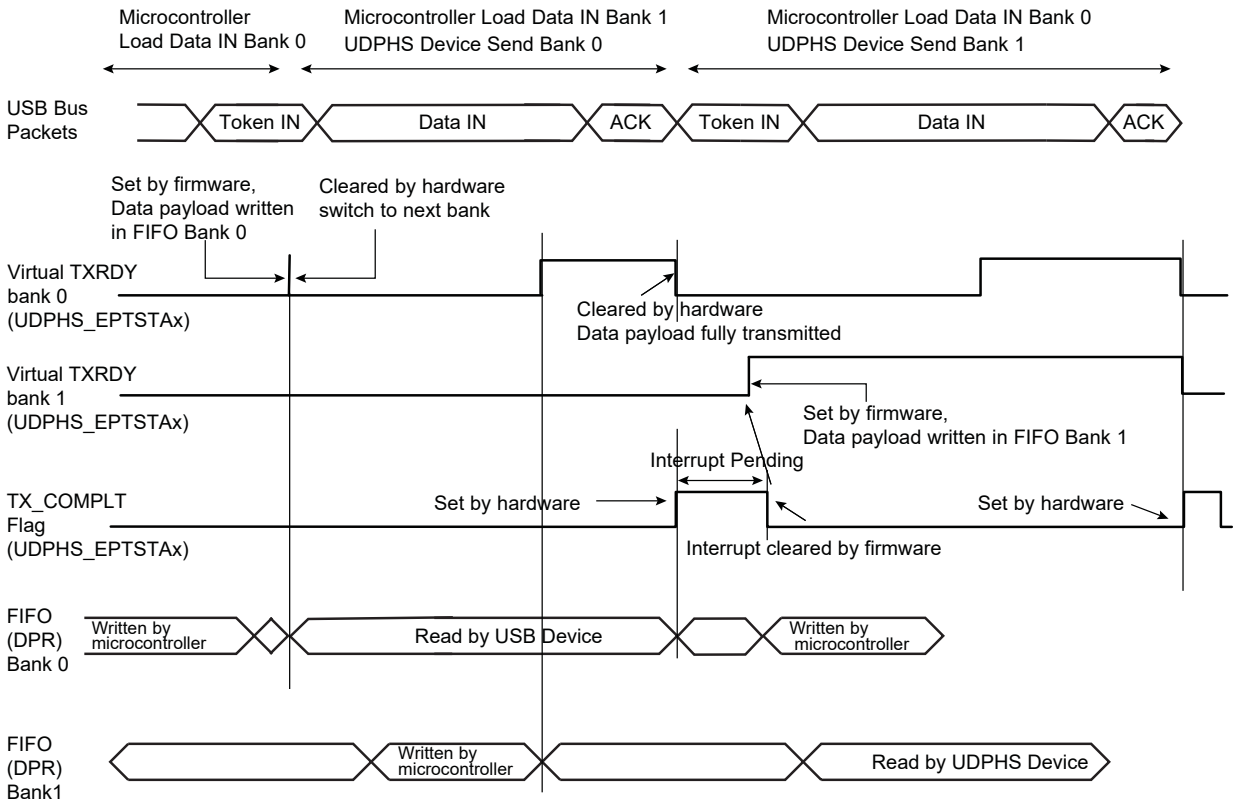
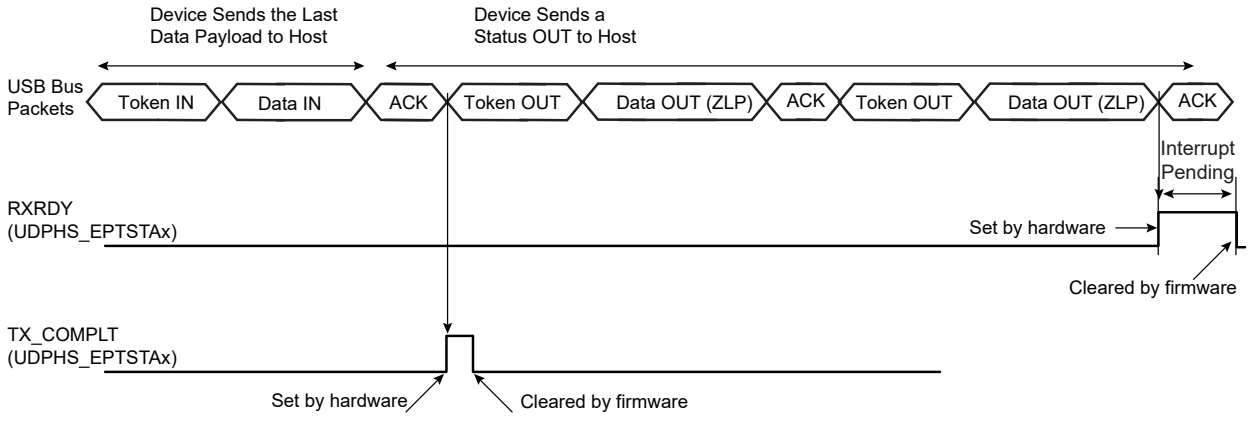
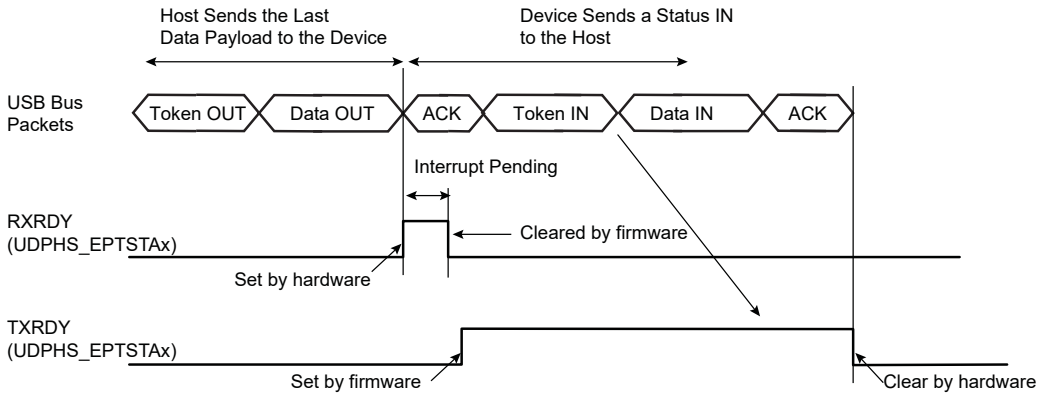



**Figure 40-11. Data IN Followed By Status OUT Transfer at the End of a Control Transfer**



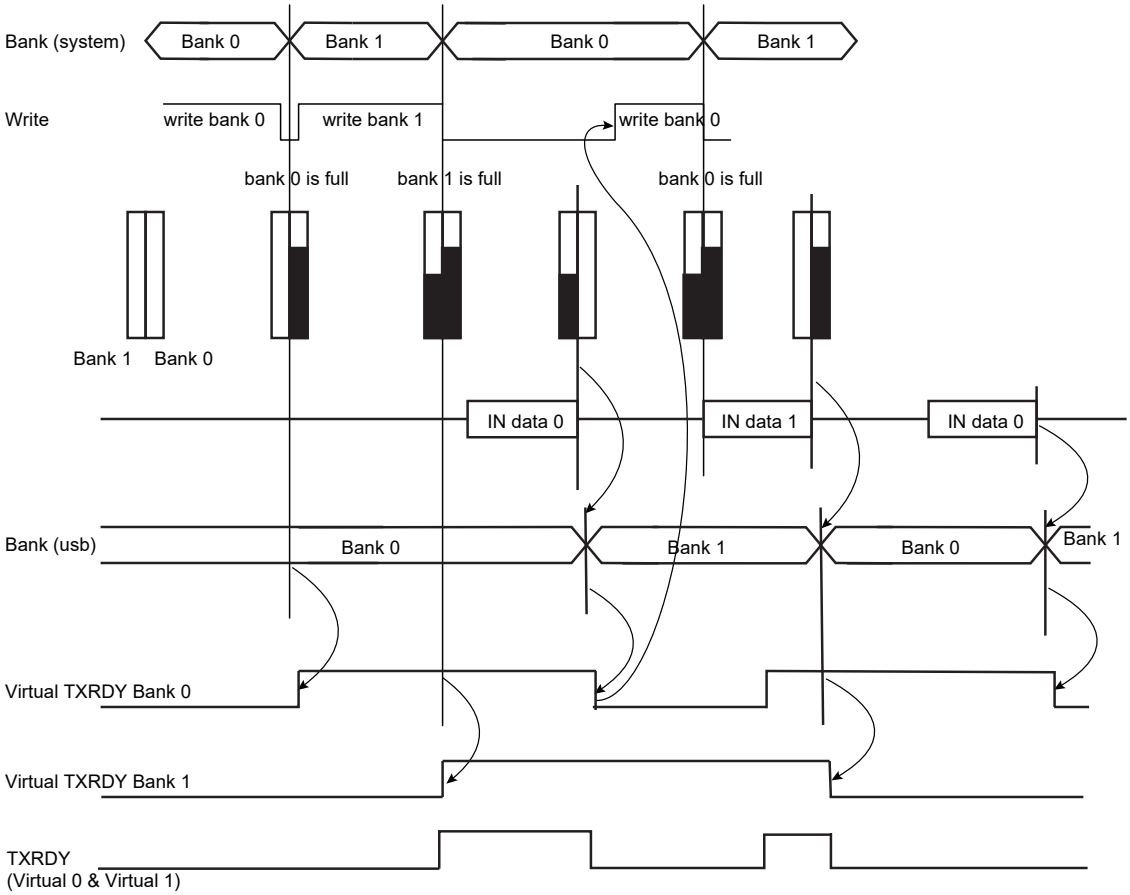

**Note:** A NAK handshake is always generated at the first status stage token.

**Figure 40-12. Data OUT Followed by Status IN Transfer**




**Note:** Before proceeding to the status stage, the software should determine that there is no risk of extra data from the host (data stage). If not certain (non-predictable data stage length), then the software should wait for a NAK-IN interrupt before proceeding to the status stage. This precaution should be taken to avoid collision in the FIFO.

**Figure 40-13. Autovalid with DMA**




**Note:** In the illustration above Autovalid validates a bank as full, although this might not be the case, in order to continue processing data and to send to DMA.

• Isochronous IN

Isochronous-IN is used to transmit a stream of data whose timing is implied by the delivery rate. Isochronous transfer provides periodic, continuous communication between host and device.

It guarantees bandwidth and low latencies appropriate for telephony, audio, video, etc.

If the endpoint is not available (TXRDY_TRER = 0), then the device does not answer to the host. An ERR_FL_ISO interrupt is generated in the UDPHS_EPTSTAx register and once enabled, then sent to the CPU.

The STALL_SNT command bit is not used for an ISO-IN endpoint.

• High Bandwidth Isochronous Endpoint Handling: IN Example

For high bandwidth isochronous endpoints, the DMA can be programmed with the number of transactions (BUFF_LENGTH field in UDPHS_DMACONTROLx) and the system should provide the required number of packets per microframe, otherwise, the host will notice a sequencing problem.

A response should be made to the first token IN recognized inside a microframe under the following conditions:

- If at least one bank has been validated, the correct DATAx corresponding to the programmed Number Of Transactions per Microframe (NB_TRANS) should be answered. In case of a subsequent missed or corrupted token IN inside the microframe, the USB 2.0 Core available data bank(s) that should normally have been transmitted during that microframe shall be flushed at its end. If this flush occurs, an error condition is flagged (ERR_FLUSH is set in UDPHS_EPTSTAx).
- If no bank is validated yet, the default DATA0 ZLP is answered and underflow is flagged (ERR_FL_ISO is set in UDPHS_EPTSTAx). Then, no data bank is flushed at microframe end.
- If no data bank has been validated at the time when a response should be made for the second transaction of NB_TRANS = 3 transactions microframe, a DATA1 ZLP is answered and underflow is flagged (ERR_FL_ISO is set in UDPHS_EPTSTAx). If and only if remaining untransmitted banks for that microframe are available at its end, they are flushed and an error condition is flagged (ERR_FLUSH is set in UDPHS_EPTSTAx).
- If no data bank has been validated at the time when a response should be made for the last programmed transaction of a microframe, a DATA0 ZLP is answered and underflow is flagged (ERR_FL_ISO is set in UDPHS_EPTSTAx). If and only if the remaining untransmitted data bank for that microframe is available at its end, it is flushed and an error condition is flagged (ERR_FLUSH is set in UDPHS_EPTSTAx).
- If at the end of a microframe no valid token IN has been recognized, no data bank is flushed and no error condition is reported.

At the end of a microframe in which at least one data bank has been transmitted, if less than NB_TRANS banks have been validated for that microframe, an error condition is flagged (ERR_TRANS is set in UDPHS_EPTSTAx).

Cases of Error (in UDPHS_EPTSTAx)

- ERR_FL_ISO: There was no data to transmit inside a microframe, so a ZLP is answered by default.
- ERR_FLUSH: At least one packet has been sent inside the microframe, but the number of token INs received is less than the number of transactions actually validated (TXRDY_TRER) and likewise with the NB_TRANS programmed.

- ERR_TRANS: At least one packet has been sent inside the microframe, but the number of token INs received is less than the number of programmed NB_TRANS transactions and the packets not requested were not validated.
- ERR_FL_ISO + ERR_FLUSH: At least one packet has been sent inside the microframe, but the data has not been validated in time to answer one of the following token INs.
- ERR_FL_ISO + ERR_TRANS: At least one packet has been sent inside the microframe, but the data has not been validated in time to answer one of the following token INs and the data can be discarded at the microframe end.
- ERR_FLUSH + ERR_TRANS: The first token IN has been answered and it was the only one received, a second bank has been validated but not the third, whereas NB_TRANS was waiting for three transactions.
- ERR_FL_ISO + ERR_FLUSH + ERR_TRANS: The first token IN has been treated, the data for the second Token IN was not available in time, but the second bank has been validated before the end of the microframe. The third bank has not been validated, but three transactions have been set in NB_TRANS.

#### 40.6.10.4 Data OUT

• Bulk OUT or Interrupt OUT

Like data IN, data OUT packets are sent by the host during the data or the status stage of control transfer or during an interrupt/bulk/isochronous OUT transfer. Data buffers are sent packet by packet under the control of the application or under the control of the DMA channel.

• Bulk OUT or Interrupt OUT: Receiving a Packet Under Application Control (Host to Device)

**Algorithm Description for Each Packet:**

- The application enables an interrupt on RXRDY_TXKL.
- When an interrupt on RXRDY_TXKL is received, the application knows that UDPHS_EPTSTAx register BYTE_COUNT bytes have been received.
- The application reads the BYTE_COUNT bytes from the endpoint.
- The application clears RXRDY_TXKL.
  **Note:** If the application does not know the size of the transfer, it may not be a good option to use AUTO_VALID. Because if a zero-length-packet is received, the RXRDY_TXKL is automatically cleared by the AUTO_VALID hardware and if the endpoint interrupt is triggered, the software will not find its originating flag when reading the UDPHS_EPTSTAx register.

**Algorithm to Fill Several Packets**

- The application enables the interrupts of BUSY_BANK and AUTO_VALID.
- When a BUSY_BANK interrupt is received, the application knows that all banks available for the endpoint have been filled. Thus, the application can read all banks available.

If the application does not know the size of the receive buffer, instead of using the BUSY_BANK interrupt, the application must use RXRDY_TXKL.

• Bulk OUT or Interrupt OUT: Sending a Buffer Using DMA (Host To Device)

To use the DMA setting, the AUTO_VALID field is mandatory.

See Bulk IN or Interrupt IN: Sending a Buffer Using DMA (Device to Host) for more information.

DMA Configuration Example:

1. First program UDPHS_DMAADDRESSx with the address of the buffer that should be transferred.

2. Enable the interrupt of the DMA in UDPHS_IEN
3. Program the DMA Channelx Control Register:
   - Size of buffer to be sent.
   - END_B_EN: Can be used for OUT packet truncation (discarding of unbuffered packet data) at the end of DMA buffer.
   - END_BUFFIT: Generate an interrupt when BUFF_COUNT in the UDPHS_DMASTATUSx register reaches 0.
   - END_TR_EN: End of transfer enable, the UDPHS device can put an end to the current DMA transfer, in case of a short packet.
   - END_TR_IT: End of transfer interrupt enable, an interrupt is sent after the last USB packet has been transferred by the DMA, if the USB transfer ended with a short packet. (Beneficial when the receive size is unknown.)
   - CHANN_ENB: Run and stop at end of buffer.

For OUT transfer, the bank will be automatically cleared by hardware when the application has read all the bytes in the bank (the bank is empty).

**Note:**

1. When a zero-length-packet is received, RXRDY_TXKL bit in UDPHS_EPTSTAx is cleared automatically by AUTO_VALID, and the application knows of the end of buffer by the presence of the END_TR_IT.
2. If the host sends a zero-length packet, and the endpoint is free, then the device sends an ACK. No data is written in the endpoint, the RXRDY_TXKL interrupt is generated, and the BYTE_COUNT field in UDPHS_EPTSTAx is null.

**Figure 40-14. Data OUT Transfer for Endpoint with One Bank**

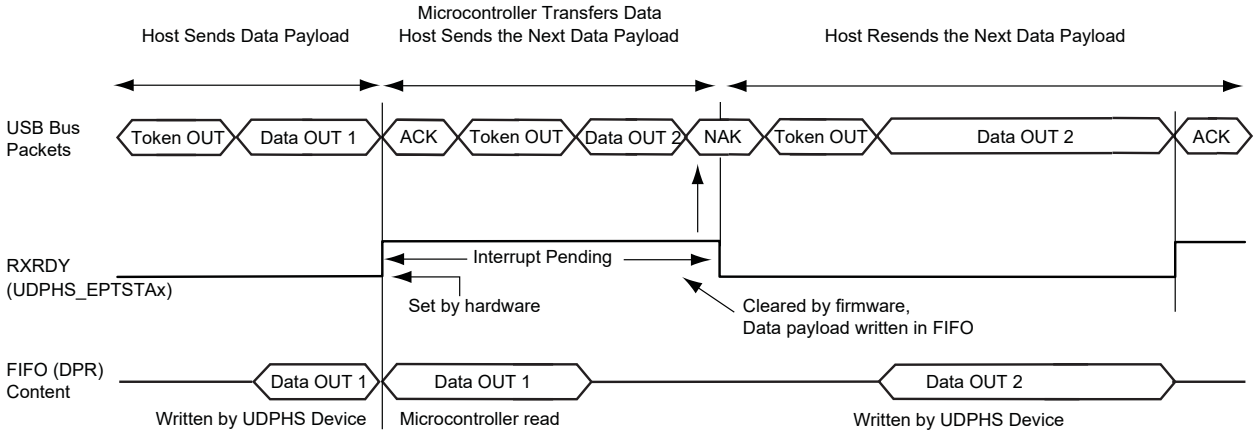
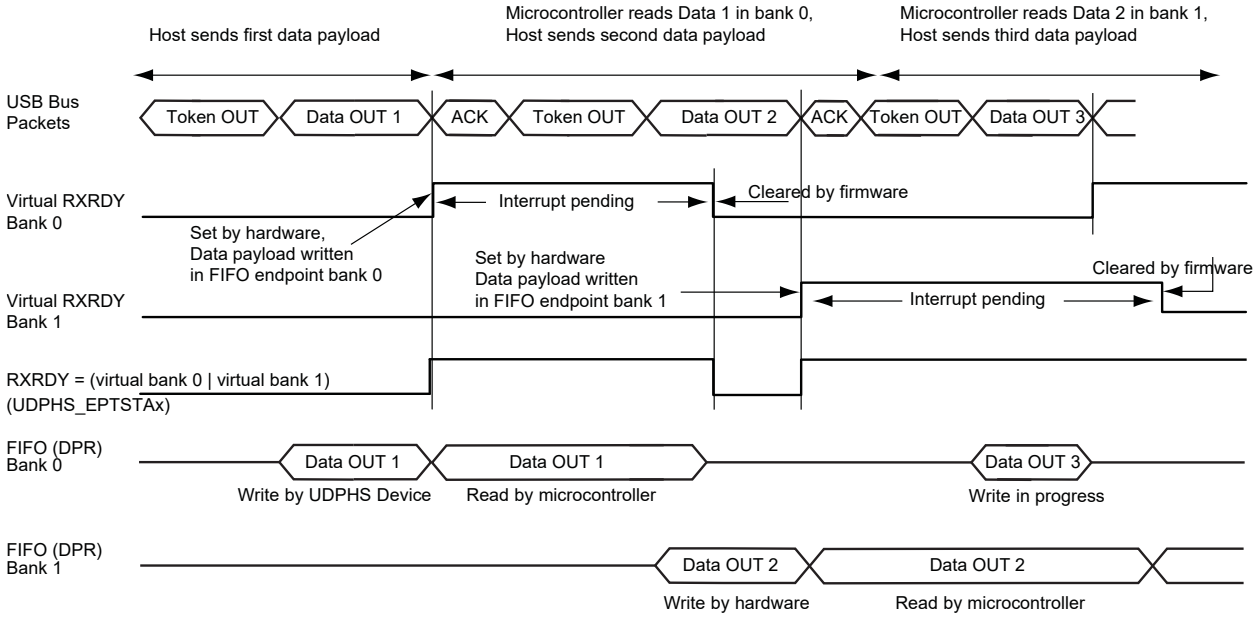


**Figure 40-15. Data OUT Transfer for an Endpoint with Two Banks**




• High Bandwidth Isochronous Endpoint OUT

USB 2.0 supports individual High Speed isochronous endpoints that require data rates up to 192 Mb/s (24 MB/s): 3x1024 data bytes per microframe.

To support such a rate, two or three banks may be used to buffer the three consecutive data packets. The microcontroller (or the DMA) should be able to empty the banks very rapidly (at least 24 MB/s on average).

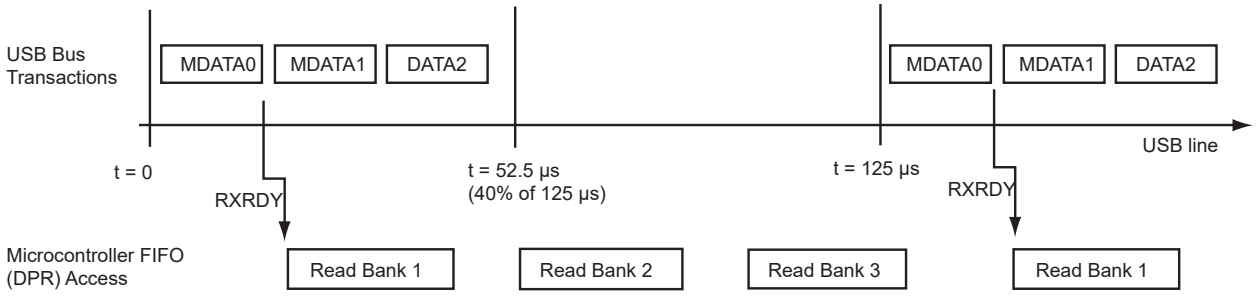
NB_TRANS field in UDPHS_EPTCFGx register = Number Of Transactions per Microframe.

If NB_TRANS > 1 then it is High Bandwidth.

Example:

- If NB_TRANS = 3, the sequence should be either
  - MData0
  - MData0/Data1
  - MData0/Data1/Data2
- If NB_TRANS = 2, the sequence should be either
  - MData0
  - MData0/Data1
- If NB_TRANS = 1, the sequence should be
  - Data0

**Figure 40-16. Bank Management, Example of Three Transactions per Microframe**




• Isochronous Endpoint Handling: OUT Example

The user can ascertain the bank status (free or busy), and the toggle sequencing of the data packet for each bank with the UDPHS_EPTSTAx register in the three fields as follows:

- TOGGLESQ_STA: PID of the data stored in the current bank
- CURBK: Number of the bank currently being accessed by the microcontroller.
- BUSY_BANK_STA: Number of busy bank

This is particularly useful in case of a missing data packet.

If the inter-packet delay between the OUT token and the Data is greater than the USB standard, then the ISO-OUT transaction is ignored. (Payload data is not written, no interrupt is generated to the CPU.)

If there is a data CRC (Cyclic Redundancy Check) error, the payload is, none the less, written in the endpoint. The ERR_CRC_NTR flag is set in UDPHS_EPTSTAx register.

If the endpoint is already full, the packet is not written in the DPRAM. The ERR_FL_ISO flag is set in UDPHS_EPTSTAx.

If the payload data is greater than the maximum size of the endpoint, then the ERR_OVFLW flag is set. It is the task of the CPU to manage this error. The data packet is written in the endpoint (except the extra data).

If the host sends a Zero Length Packet, and the endpoint is free, no data is written in the endpoint, the RXRDY_TXKL flag is set, and the BYTE_COUNT field in UDPHS_EPTSTAx register is null.

The FRCESTALL command bit is unused for an isochronous endpoint.

Otherwise, payload data is written in the endpoint, the RXRDY_TXKL interrupt is generated and the BYTE_COUNT in UDPHS_EPTSTAx register is updated.

#### 40.6.10.5 STALL

STALL is returned by a function in response to an IN token or after the data phase of an OUT or in response to a PING transaction. STALL indicates that a function is unable to transmit or receive data, or that a control pipe request is not supported.

- OUT

To stall an endpoint, set the FRCESTALL bit in UDPHS_EPTSETSTAx register and after the STALL_SNT flag has been set, set the TOGGLE_SEG bit in the UDPHS_EPTCLRSTAx register.

- IN

Set the FRCESTALL bit in UDPHS_EPTSETSTAx register.

**Figure 40-17. Stall Handshake Data OUT Transfer**

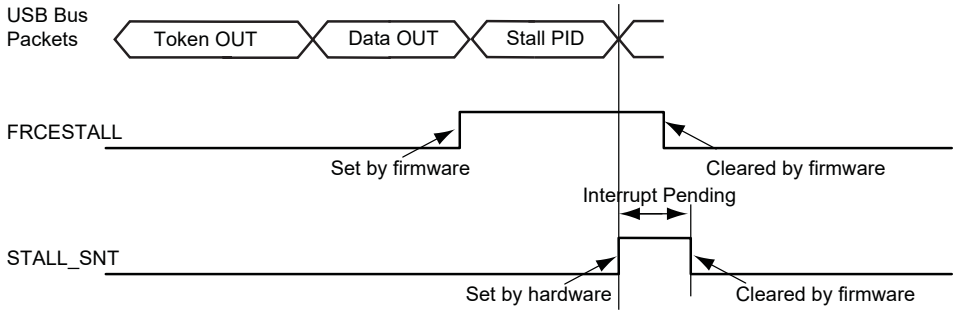
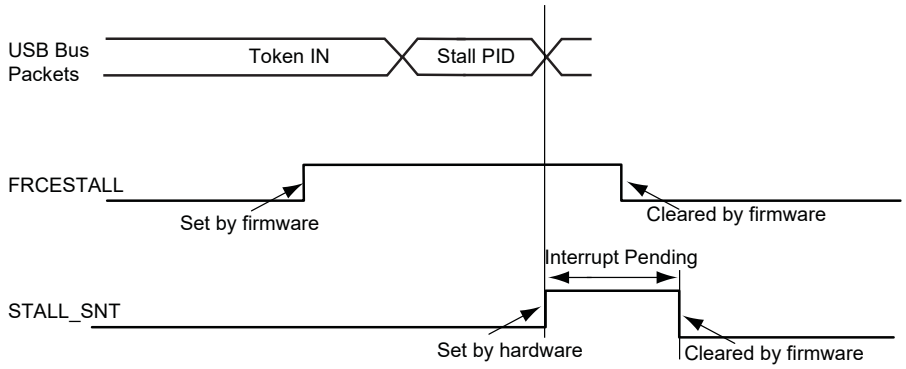



**Figure 40-18. Stall Handshake Data IN Transfer**




### 40.6.11 Speed Identification

The high speed reset is managed by hardware.

At the connection, the host makes a reset which could be a classic reset (full speed) or a high speed reset.

At the end of the reset process (full or high), the ENDRESET interrupt is generated.

Then the CPU should read the SPEED bit in UDPHS_INTSTAx to ascertain the speed mode of the device.

### 40.6.12 USB V2.0 High Speed Global Interrupt

Interrupts are defined in UDPHS Interrupt Enable Register (UDPHS_IEN) and in UDPHS Interrupt Status Register (UDPHS_INTSTA).

### 40.6.13 Endpoint Interrupts

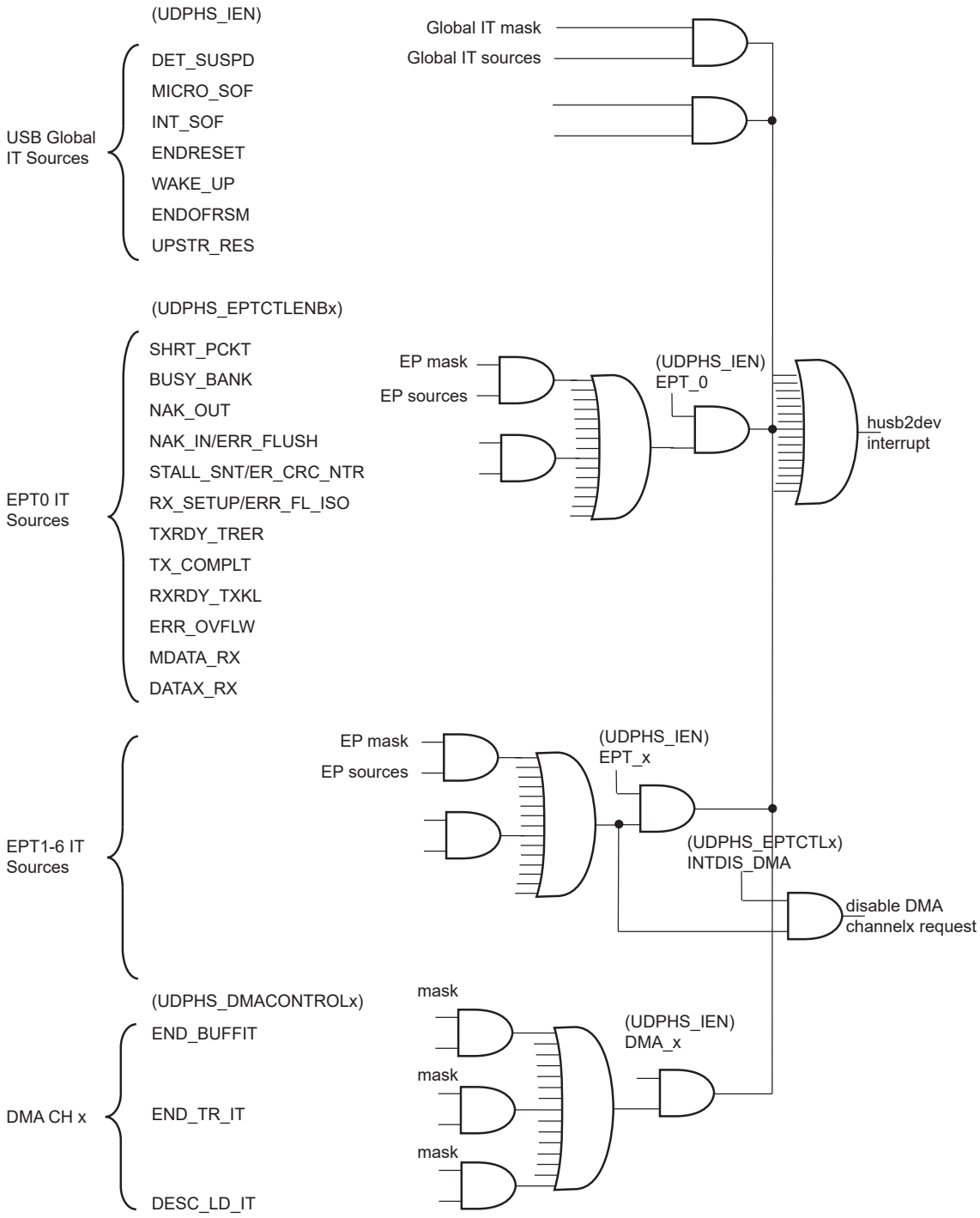
Interrupts are enabled in UDPHS_IEN (see UDPHS Interrupt Enable Register) and individually masked in UDPHS_EPTCTLENBx (see UDPHS Endpoint Control Enable Register (Control, Bulk, Interrupt Endpoints)).

**Table 40-4. Endpoint Interrupt Source Masks**

| | |
|---|---|
| SHRT_PCKT | Short Packet Interrupt |
| BUSY_BANK | Busy Bank Interrupt |
| NAK_OUT | NAKOUT Interrupt |
| NAK_IN/ERR_FLUSH | NAKIN/Error Flush Interrupt |

| | |
|---|---|
| STALL_SNT/ERR_CRC_NTR | Stall Sent/CRC error/Number of Transaction Error Interrupt |
| RX_SETUP/ERR_FL_ISO | Received SETUP/Error Flow Interrupt |
| TXRDY_TRER | TX Packet Read/Transaction Error Interrupt |
| TX_COMPLT | Transmitted IN Data Complete Interrupt |
| RXRDY_TXKL | Received OUT Data Interrupt |
| ERR_OVFLW | Overflow Error Interrupt |
| MDATA_RX | MDATA Interrupt |
| DATAX_RX | DATAx Interrupt |

**Figure 40-19. UDPHS Interrupt Control Interface**

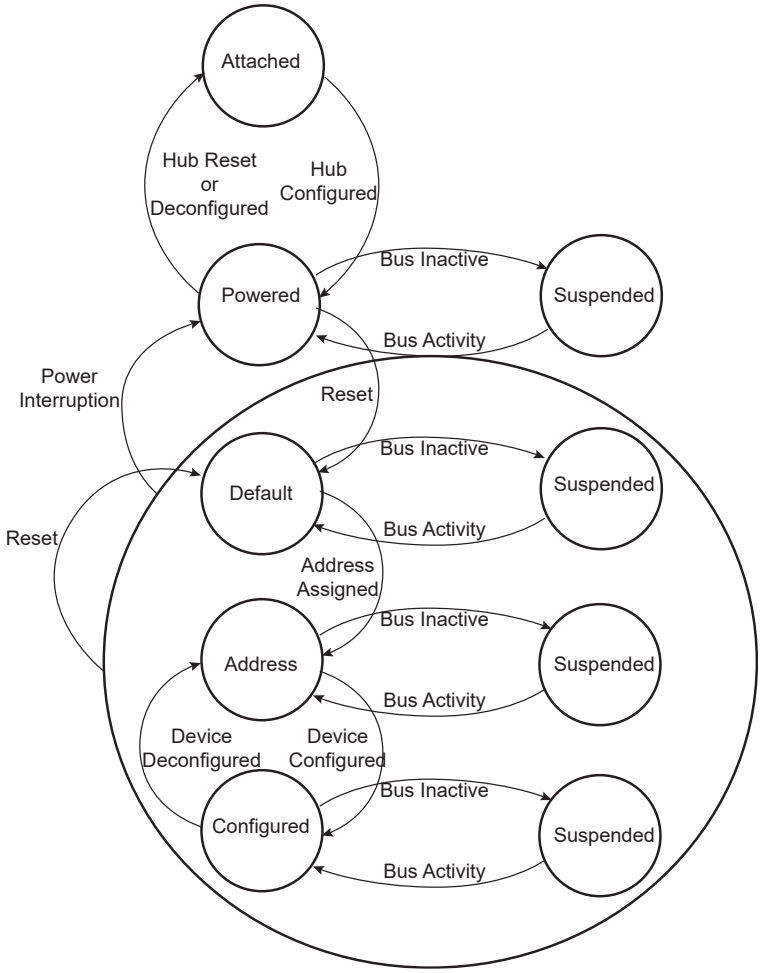



### 40.6.14 Power Modes

#### 40.6.14.1 Controlling Device States

A USB device has several possible states. Refer to Chapter 9 (USB Device Framework) of the Universal Serial Bus Specification, Rev 2.0.

**Figure 40-20. UDPHS Device State Diagram**




Movement from one state to another depends on the USB bus state or on standard requests sent through control transactions via the default endpoint (endpoint 0).

After a period of bus inactivity, the USB device enters Suspend mode. Accepting Suspend/Resume requests from the USB host is mandatory. Constraints in Suspend mode are very strict for bus-powered applications; devices may not consume more than 500 µA on the USB bus.

While in Suspend mode, the host may wake up a device by sending a resume signal (bus activity) or a USB device may send a wakeup request to the host, e.g., waking up a PC by moving a USB mouse.

The wakeup feature is not mandatory for all devices and must be negotiated with the host.

#### 40.6.14.2 Not Powered State

Self powered devices can detect 5V VBUS using a PIO. When the device is not connected to a host, device power consumption can be reduced by the DETACH bit in UDPHS_CTRL. Disabling the transceiver is automatically done. HSDM, HSDP, FSDP and FSDP lines are tied to GND pulldowns integrated in the hub downstream ports.

#### 40.6.14.3 Entering Attached State

When no device is connected, the USB FSDP and FSDM signals are tied to GND by 15 KΩ pulldowns integrated in the hub downstream ports. When a device is attached to an hub downstream port, the device connects a 1.5 KΩ pullup on FSDP. The USB bus line goes into IDLE state, FSDP is pulled up by the device 1.5 KΩ resistor to 3.3V and FSDM is pulled down by the 15 KΩ resistor to GND of the host.

After pullup connection, the device enters the powered state. The transceiver remains disabled until bus activity is detected.

In case of low power consumption need, the device can be stopped. When the device detects the VBUS, the software must enable the USB transceiver by enabling the EN_UDPHS bit in UDPHS_CTRL register.

The software can detach the pullup by setting DETACH bit in UDPHS_CTRL register.

#### 40.6.14.4 From Powered State to Default State (Reset)

After its connection to a USB host, the USB device waits for an end-of-bus reset. The unmasked flag ENDRESET is set in the UDPHS_IEN register and an interrupt is triggered.

Once the ENDRESET interrupt has been triggered, the device enters Default State. In this state, the UDPHS software must:

- Enable the default endpoint, setting the EPT_ENABL flag in the UDPHS_EPTCTLENB[0] register and, optionally, enabling the interrupt for endpoint 0 by writing 1 in EPT_0 of the UDPHS_IEN register. The enumeration then begins by a control transfer.
- Configure the Interrupt Mask Register which has been reset by the USB reset detection
- Enable the transceiver.

In this state, the EN_UDPHS bit in UDPHS_CTRL register must be enabled.

#### 40.6.14.5 From Default State to Address State (Address Assigned)

After a Set Address standard device request, the USB host peripheral enters the address state.

**⚠WARNING** Before the device enters address state, it must achieve the Status IN transaction of the control transfer, i.e., the UDPHS device sets its new address once the TX_COMPLT flag in the UDPHS_EPTCTL[0] register has been received and cleared.

To move to address state, the driver software sets the DEV_ADDR field and the FADDR_EN flag in the UDPHS_CTRL register.

#### 40.6.14.6 From Address State to Configured State (Device Configured)

Once a valid Set Configuration standard request has been received and acknowledged, the device enables endpoints corresponding to the current configuration. This is done by setting the BK_NUMBER, EPT_TYPE, EPT_DIR and EPT_SIZE fields in the UDPHS_EPTCFGx registers and enabling them by setting the EPT_ENABL flag in the UDPHS_EPTCTLENBx registers, and, optionally, enabling corresponding interrupts in the UDPHS_IEN register.

#### 40.6.14.7 Entering Suspend State (Bus Activity)

When a Suspend (no bus activity on the USB bus) is detected, the DET_SUSPD signal in the UDPHS_STA register is set. This triggers an interrupt if the corresponding bit is set in the UDPHS_IEN register. This flag is cleared by writing to the UDPHS_CLRINT register. Then the device enters Suspend mode.

In this state bus powered devices must drain less than 500 µA from the 5V VBUS. As an example, the microcontroller switches to slow clock, disables the PLL and main oscillator, and goes into Idle mode. It may also switch off other devices on the board.

The UDPHS device peripheral clocks can be switched off. Resume event is asynchronously detected.

#### 40.6.14.8 Receiving a Host Resume

In Suspend mode, a resume event on the USB bus line is detected asynchronously, transceiver and clocks disabled (however, the pullup should not be removed).

Once the resume is detected on the bus, the signal WAKE_UP in the UDPHS_INTSTA is set. It may generate an interrupt if the corresponding bit in the UDPHS_IEN register is set. This interrupt may be used to wake up the core, enable PLL and main oscillators and configure clocks.

#### 40.6.14.9 Sending an External Resume

In Suspend State it is possible to wake up the host by sending an external resume.

The device waits at least 5 ms after being entered in Suspend State before sending an external resume.

The device must force a K state from 1 to 15 ms to resume the host.

### 40.6.15 Test Mode

A device must support the TEST_MODE feature when in the Default, Address or Configured High Speed device states.

TEST_MODE can be:

- Test_J
- Test_K
- Test_Packet
- Test_SEO_NAK

(See UDPHS Test Register for definitions of each test mode.)

```
const char test_packet_buffer[] = {
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,                // JKJKJKJK * 9
0xAA,0xAA,0xAA,0xAA,0xAA,0xAA,0xAA,0xAA,                     // JJKKJJKK * 8
0xEE,0xEE,0xEE,0xEE,0xEE,0xEE,0xEE,0xEE,                     // JJKKJJKK * 8
0xFE,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF, // JJJJJJJKKKKKKK * 8
0x7F,0xBF,0xDF,0xEF,0xF7,0xFB,0xFD,                          // JJJJJJJK * 8
0xFC,0x7E,0xBF,0xDF,0xEF,0xF7,0xFB,0xFD,0x7E                 // {JKKKKKKK * 10}, JK
};
```

## 40.7 Register Summary

**Note:**
The registers below have two modes: Control, Bulk, Interrupt Endpoints mode and Isochronous Endpoints mode.

In this register summary, both modes are displayed at the same offset.

- UDPHS_EPTCTLENB
- UDPHS_EPTCTLDIS
- UDPHS_EPTCTL
- UDPHS_EPTSETSTA
- UDPHS_EPTCLRSTA
- UDPHS_EPTSTA

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | UDPHS_CTRL | 7:0 | FADDR_EN | DEV_ADDR[6:0] | | | | | | |
| | | 15:8 | | | | | PULLD_DIS | REWAKEUP | DETACH | EN_UDPHS |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | UDPHS_FNUM | 7:0 | FRAME_NUMBER[4:0] | | | | | MICRO_FRAME_NUM[2:0] | | |
| | | 15:8 | | | FRAME_NUMBER[10:5] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | FNUM_ERR | | | | | | | |
| 0x08 ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | UDPHS_IEN | 7:0 | UPSTR_RES | ENDOFRSM | WAKE_UP | ENDRESET | INT_SOF | MICRO_SOF | DET_SUSPD | |
| | | 15:8 | EPT_7 | EPT_6 | EPT_5 | EPT_4 | EPT_3 | EPT_2 | EPT_1 | EPT_0 |
| | | 23:16 | EPT_15 | EPT_14 | EPT_13 | EPT_12 | EPT_11 | EPT_10 | EPT_9 | EPT_8 |
| | | 31:24 | DMA_7 | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | |
| 0x14 | UDPHS_INTSTA | 7:0 | UPSTR_RES | ENDOFRSM | WAKE_UP | ENDRESET | INT_SOF | MICRO_SOF | DET_SUSPD | SPEED |
| | | 15:8 | EPT_7 | EPT_6 | EPT_5 | EPT_4 | EPT_3 | EPT_2 | EPT_1 | EPT_0 |
| | | 23:16 | EPT_15 | EPT_14 | EPT_13 | EPT_12 | EPT_11 | EPT_10 | EPT_9 | EPT_8 |
| | | 31:24 | DMA_7 | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | |
| 0x18 | UDPHS_CLRINT | 7:0 | UPSTR_RES | ENDOFRSM | WAKE_UP | ENDRESET | INT_SOF | MICRO_SOF | DET_SUSPD | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | UDPHS_EPTRST | 7:0 | EPT_7 | EPT_6 | EPT_5 | EPT_4 | EPT_3 | EPT_2 | EPT_1 | EPT_0 |
| | | 15:8 | EPT_15 | EPT_14 | EPT_13 | EPT_12 | EPT_11 | EPT_10 | EPT_9 | EPT_8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 ... 0xDF | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xE0 | UDPHS_TST | 7:0 | | | OPMODE2 | TST_PKT | TST_K | TST_J | SPEED_CFG[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xE4 ... 0xFF | Reserved | | | | | | | | | |
| 0x0100 | UDPHS_EPTCFG0 | 7:0 | BK_NUMBER[1:0] | | EPT_TYPE[1:0] | | EPT_DIR | EPT_SIZE[2:0] | | |
| | | 15:8 | | | | | | | NB_TRANS[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | EPT_MAPD | | | | | | | |
| 0x0104 | UDPHS_EPTCTLENB0 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0104 | UDPHS_EPTCTLENB0 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0108 | UDPHS_EPTCTLDIS0 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0108 | UDPHS_EPTCTLDIS0 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x010C | UDPHS_EPTCTL0 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x010C | UDPHS_EPTCTL0 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0110 ... 0x0113 | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0114 | UDPHS_EPTSETSTA0 | 7:0 | | | FRCESTALL | | | | | |
| | | 15:8 | | | | | TXRDY | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0114 | UDPHS_EPTSETSTA0 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | TXRDY_TRER | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0118 | UDPHS_EPTCLRSTA0 | 7:0 | | TOGGLESQ | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0118 | UDPHS_EPTCLRSTA0 | 7:0 | | TOGGLESQ | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x011C | UDPHS_EPTSTA0 | 7:0 | TOGGLESQ_STA[1:0] | | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK_CTLDIR[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x011C | UDPHS_EPTSTA0 | 7:0 | TOGGLESQ_STA[1:0] | | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x0120 | UDPHS_EPTCFG1 | 7:0 | BK_NUMBER[1:0] | | EPT_TYPE[1:0] | | EPT_DIR | EPT_SIZE[2:0] | | |
| | | 15:8 | | | | | | | NB_TRANS[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | EPT_MAPD | | | | | | | |
| 0x0124 | UDPHS_EPTCTLENB1 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0124 | UDPHS_EPTCTLENB1 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0128 | UDPHS_EPTCTLDIS1 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0128 | UDPHS_EPTCTLDIS1 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x012C | UDPHS_EPTCTL1 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x012C | UDPHS_EPTCTL1 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0130 ... 0x0133 | Reserved | | | | | | | | | |
| 0x0134 | UDPHS_EPTSETSTA1 | 7:0 | | | FRCESTALL | | | | | |
| | | 15:8 | | | | | TXRDY | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0134 | UDPHS_EPTSETSTA1 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | TXRDY_TRER | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0138 | UDPHS_EPTCLRSTA1 | 7:0 | | TOGGLESQ | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0138 | UDPHS_EPTCLRSTA1 | 7:0 | | TOGGLESQ | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x013C | UDPHS_EPTSTA1 | 7:0 | TOGGLESQ_STA[1:0] | | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK_CTLDIR[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x013C | UDPHS_EPTSTA1 | 7:0 | TOGGLESQ_STA[1:0] | | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x0140 | UDPHS_EPTCFG2 | 7:0 | BK_NUMBER[1:0] | | EPT_TYPE[1:0] | | EPT_DIR | EPT_SIZE[2:0] | | |
| | | 15:8 | | | | | | | NB_TRANS[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | EPT_MAPD | | | | | | | |
| 0x0144 | UDPHS_EPTCTLENB2 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0144 | UDPHS_EPTCTLENB2 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0148 | UDPHS_EPTCTLDIS2 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0148 | UDPHS_EPTCTLDIS2 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x014C | UDPHS_EPTCTL2 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x014C | UDPHS_EPTCTL2 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0150 ... 0x0153 | Reserved | | | | | | | | | |
| 0x0154 | UDPHS_EPTSETSTA2 | 7:0 | | | FRCESTALL | | | | | |
| | | 15:8 | | | | | TXRDY | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0154 | UDPHS_EPTSETSTA2 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | TXRDY_TRER | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0158 | UDPHS_EPTCLRSTA2 | 7:0 | | TOGGLESQ | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0158 | UDPHS_EPTCLRSTA2 | 7:0 | | TOGGLESQ | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x015C | UDPHS_EPTSTA2 | 7:0 | TOGGLESQ_STA[1:0] | | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK_CTLDIR[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x015C | UDPHS_EPTSTA2 | 7:0 | TOGGLESQ_STA[1:0] | | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x0160 | UDPHS_EPTCFG3 | 7:0 | BK_NUMBER[1:0] | | EPT_TYPE[1:0] | | EPT_DIR | EPT_SIZE[2:0] | | |
| | | 15:8 | | | | | | | NB_TRANS[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | EPT_MAPD | | | | | | | |
| 0x0164 | UDPHS_EPTCTLENB3 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0164 | UDPHS_EPTCTLENB3 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0168 | UDPHS_EPTCTLDIS3 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0168 | UDPHS_EPTCTLDIS3 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x016C | UDPHS_EPTCTL3 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x016C | UDPHS_EPTCTL3 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0170 ... 0x0173 | Reserved | | | | | | | | | |
| 0x0174 | UDPHS_EPTSETSTA3 | 7:0 | | | FRCESTALL | | | | | |
| | | 15:8 | | | | | TXRDY | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0174 | UDPHS_EPTSETSTA3 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | TXRDY_TRER | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0178 | UDPHS_EPTCLRSTA3 | 7:0 | | TOGGLESQ | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0178 | UDPHS_EPTCLRSTA3 | 7:0 | | TOGGLESQ | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x017C | UDPHS_EPTSTA3 | 7:0 | TOGGLESQ_STA[1:0] | | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK_CTLDIR[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x017C | UDPHS_EPTSTA3 | 7:0 | TOGGLESQ_STA[1:0] | | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x0180 | UDPHS_EPTCFG4 | 7:0 | BK_NUMBER[1:0] | | EPT_TYPE[1:0] | | EPT_DIR | EPT_SIZE[2:0] | | |
| | | 15:8 | | | | | | | NB_TRANS[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | EPT_MAPD | | | | | | | |
| 0x0184 | UDPHS_EPTCTLENB4 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0184 | UDPHS_EPTCTLENB4 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0188 | UDPHS_EPTCTLDIS4 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0188 | UDPHS_EPTCTLDIS4 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x018C | UDPHS_EPTCTL4 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x018C | UDPHS_EPTCTL4 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0190 ... 0x0193 | Reserved | | | | | | | | | |
| 0x0194 | UDPHS_EPTSETSTA4 | 7:0 | | | FRCESTALL | | | | | |
| | | 15:8 | | | | | TXRDY | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0194 | UDPHS_EPTSETSTA4 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | TXRDY_TRER | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0198 | UDPHS_EPTCLRSTA4 | 7:0 | | TOGGLESQ | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0198 | UDPHS_EPTCLRSTA4 | 7:0 | | TOGGLESQ | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x019C | UDPHS_EPTSTA4 | 7:0 | TOGGLESQ_STA[1:0] | | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK_CTLDIR[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x019C | UDPHS_EPTSTA4 | 7:0 | TOGGLESQ_STA[1:0] | | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x01A0 | UDPHS_EPTCFG5 | 7:0 | BK_NUMBER[1:0] | | EPT_TYPE[1:0] | | EPT_DIR | EPT_SIZE[2:0] | | |
| | | 15:8 | | | | | | | NB_TRANS[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | EPT_MAPD | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x01A4 | UDPHS_EPTCTLENB5 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x01A4 | UDPHS_EPTCTLENB5 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x01A8 | UDPHS_EPTCTLDIS5 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x01A8 | UDPHS_EPTCTLDIS5 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x01AC | UDPHS_EPTCTL5 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x01AC | UDPHS_EPTCTL5 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x01B0 ... 0x01B3 | Reserved | | | | | | | | | |
| 0x01B4 | UDPHS_EPTSETSTA5 | 7:0 | | | FRCESTALL | | | | | |
| | | 15:8 | | | | | TXRDY | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01B4 | UDPHS_EPTSETSTA5 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | TXRDY_TRER | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x01B8 | UDPHS_EPTCLRSTA5 | 7:0 | | TOGGLESQ | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01B8 | UDPHS_EPTCLRSTA5 | 7:0 | | TOGGLESQ | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01BC | UDPHS_EPTSTA5 | 7:0 | TOGGLESQ_STA[1:0] | | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK_CTLDIR[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x01BC | UDPHS_EPTSTA5 | 7:0 | TOGGLESQ_STA[1:0] | | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x01C0 | UDPHS_EPTCFG6 | 7:0 | BK_NUMBER[1:0] | | EPT_TYPE[1:0] | | EPT_DIR | EPT_SIZE[2:0] | | |
| | | 15:8 | | | | | | | NB_TRANS[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | EPT_MAPD | | | | | | | |
| 0x01C4 | UDPHS_EPTCTLENB6 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x01C4 | UDPHS_EPTCTLENB6 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x01C8 | UDPHS_EPTCTLDIS6 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x01C8 | UDPHS_EPTCTLDIS6 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x01CC | UDPHS_EPTCTL6 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x01CC | UDPHS_EPTCTL6 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x01D0 ... 0x01D3 | Reserved | | | | | | | | | |
| 0x01D4 | UDPHS_EPTSETSTA6 | 7:0 | | | FRCESTALL | | | | | |
| | | 15:8 | | | | | TXRDY | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01D4 | UDPHS_EPTSETSTA6 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | TXRDY_TRER | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01D8 | UDPHS_EPTCLRSTA6 | 7:0 | | TOGGLESQ | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01D8 | UDPHS_EPTCLRSTA6 | 7:0 | | TOGGLESQ | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01DC | UDPHS_EPTSTA6 | 7:0 | TOGGLESQ_STA[1:0] | | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK_CTLDIR[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x01DC | UDPHS_EPTSTA6 | 7:0 | TOGGLESQ_STA[1:0] | | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x01E0 | UDPHS_EPTCFG7 | 7:0 | BK_NUMBER[1:0] | | EPT_TYPE[1:0] | | EPT_DIR | EPT_SIZE[2:0] | | | |
| | | 15:8 | | | | | | | NB_TRANS[1:0] | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | EPT_MAPD | | | | | | | | |
| 0x01E4 | UDPHS_EPTCTLENB7 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW | |
| | | 23:16 | | | | | | BUSY_BANK | | | |
| | | 31:24 | SHRT_PCKT | | | | | | | | |
| 0x01E4 | UDPHS_EPTCTLENB7 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW | |
| | | 23:16 | | | | | | BUSY_BANK | | | |
| | | 31:24 | SHRT_PCKT | | | | | | | | |
| 0x01E8 | UDPHS_EPTCTLDIS7 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_DISABL | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW | |
| | | 23:16 | | | | | | BUSY_BANK | | | |
| | | 31:24 | SHRT_PCKT | | | | | | | | |
| 0x01E8 | UDPHS_EPTCTLDIS7 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_DISABL | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW | |
| | | 23:16 | | | | | | BUSY_BANK | | | |
| | | 31:24 | SHRT_PCKT | | | | | | | | |
| 0x01EC | UDPHS_EPTCTL7 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW | |
| | | 23:16 | | | | | | BUSY_BANK | | | |
| | | 31:24 | SHRT_PCKT | | | | | | | | |
| 0x01EC | UDPHS_EPTCTL7 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW | |
| | | 23:16 | | | | | | BUSY_BANK | | | |
| | | 31:24 | SHRT_PCKT | | | | | | | | |
| 0x01F0 ... 0x01F3 | Reserved | | | | | | | | | | |
| 0x01F4 | UDPHS_EPTSETSTA7 | 7:0 | | | FRCESTALL | | | | | | |
| | | 15:8 | | | | | TXRDY | | RXRDY_TXKL | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x01F4 | UDPHS_EPTSETSTA7 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | TXRDY_TRER | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01F8 | UDPHS_EPTCLRSTA7 | 7:0 | | TOGGLESQ | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01F8 | UDPHS_EPTCLRSTA7 | 7:0 | | TOGGLESQ | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01FC | UDPHS_EPTSTA7 | 7:0 | TOGGLESQ_STA[1:0] | | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK_CTLDIR[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x01FC | UDPHS_EPTSTA7 | 7:0 | TOGGLESQ_STA[1:0] | | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x0200 | UDPHS_EPTCFG8 | 7:0 | BK_NUMBER[1:0] | | EPT_TYPE[1:0] | | EPT_DIR | EPT_SIZE[2:0] | | |
| | | 15:8 | | | | | | | NB_TRANS[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | EPT_MAPD | | | | | | | |
| 0x0204 | UDPHS_EPTCTLENB8 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0204 | UDPHS_EPTCTLENB8 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0208 | UDPHS_EPTCTLDIS8 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0208 | UDPHS_EPTCTLDIS8 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x020C | UDPHS_EPTCTL8 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x020C | UDPHS_EPTCTL8 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0210 ... 0x0213 | Reserved | | | | | | | | | |
| 0x0214 | UDPHS_EPTSETSTA8 | 7:0 | | | FRCESTALL | | | | | |
| | | 15:8 | | | | | TXRDY | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0214 | UDPHS_EPTSETSTA8 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | TXRDY_TRER | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0218 | UDPHS_EPTCLRSTA8 | 7:0 | | TOGGLESQ | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0218 | UDPHS_EPTCLRSTA8 | 7:0 | | TOGGLESQ | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x021C | UDPHS_EPTSTA8 | 7:0 | TOGGLESQ_STA[1:0] | | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK_CTLDIR[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x021C | UDPHS_EPTSTA8 | 7:0 | TOGGLESQ_STA[1:0] | | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x0220 | UDPHS_EPTCFG9 | 7:0 | BK_NUMBER[1:0] | | EPT_TYPE[1:0] | | EPT_DIR | EPT_SIZE[2:0] | | |
| | | 15:8 | | | | | | | NB_TRANS[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | EPT_MAPD | | | | | | | |
| 0x0224 | UDPHS_EPTCTLENB9 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0224 | UDPHS_EPTCTLENB9 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0228 | UDPHS_EPTCTLDIS9 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0228 | UDPHS_EPTCTLDIS9 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x022C | UDPHS_EPTCTL9 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x022C | UDPHS_EPTCTL9 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0230 ... 0x0233 | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0234 | UDPHS_EPTSETSTA9 | 7:0 | | | FRCESTALL | | | | | |
| | | 15:8 | | | | | TXRDY | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0234 | UDPHS_EPTSETSTA9 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | TXRDY_TRER | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0238 | UDPHS_EPTCLRSTA9 | 7:0 | | TOGGLESQ | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0238 | UDPHS_EPTCLRSTA9 | 7:0 | | TOGGLESQ | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x023C | UDPHS_EPTSTA9 | 7:0 | TOGGLESQ_STA[1:0] | | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK_CTLDIR[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x023C | UDPHS_EPTSTA9 | 7:0 | TOGGLESQ_STA[1:0] | | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x0240 | UDPHS_EPTCFG10 | 7:0 | BK_NUMBER[1:0] | | EPT_TYPE[1:0] | | EPT_DIR | EPT_SIZE[2:0] | | |
| | | 15:8 | | | | | | | NB_TRANS[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | EPT_MAPD | | | | | | | |
| 0x0244 | UDPHS_EPTCTLENB10 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0244 | UDPHS_EPTCTLENB10 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0248 | UDPHS_EPTCTLDIS10 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0248 | UDPHS_EPTCTLDIS10 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x024C | UDPHS_EPTCTL10 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x024C | UDPHS_EPTCTL10 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0250 ... 0x0253 | Reserved | | | | | | | | | |
| 0x0254 | UDPHS_EPTSETSTA10 | 7:0 | | | FRCESTALL | | | | | |
| | | 15:8 | | | | | TXRDY | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0254 | UDPHS_EPTSETSTA10 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | TXRDY_TRER | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0258 | UDPHS_EPTCLRSTA10 | 7:0 | | TOGGLESQ | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0258 | UDPHS_EPTCLRSTA10 | 7:0 | | TOGGLESQ | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x025C | UDPHS_EPTSTA10 | 7:0 | TOGGLESQ_STA[1:0] | | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK_CTLDIR[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x025C | UDPHS_EPTSTA10 | 7:0 | TOGGLESQ_STA[1:0] | | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x0260 | UDPHS_EPTCFG11 | 7:0 | BK_NUMBER[1:0] | | EPT_TYPE[1:0] | | EPT_DIR | EPT_SIZE[2:0] | | |
| | | 15:8 | | | | | | | NB_TRANS[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | EPT_MAPD | | | | | | | |
| 0x0264 | UDPHS_EPTCTLENB11 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0264 | UDPHS_EPTCTLENB11 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0268 | UDPHS_EPTCTLDIS11 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0268 | UDPHS_EPTCTLDIS11 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x026C | UDPHS_EPTCTL11 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x026C | UDPHS_EPTCTL11 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0270 ... 0x0273 | Reserved | | | | | | | | | |
| 0x0274 | UDPHS_EPTSETSTA11 | 7:0 | | | FRCESTALL | | | | | |
| | | 15:8 | | | | | TXRDY | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0274 | UDPHS_EPTSETSTA11 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | TXRDY_TRER | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0278 | UDPHS_EPTCLRSTA11 | 7:0 | | TOGGLESQ | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0278 | UDPHS_EPTCLRSTA11 | 7:0 | | TOGGLESQ | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x027C | UDPHS_EPTSTA11 | 7:0 | TOGGLESQ_STA[1:0] | | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK_CTLDIR[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x027C | UDPHS_EPTSTA11 | 7:0 | TOGGLESQ_STA[1:0] | | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x0280 | UDPHS_EPTCFG12 | 7:0 | BK_NUMBER[1:0] | | EPT_TYPE[1:0] | | EPT_DIR | EPT_SIZE[2:0] | | |
| | | 15:8 | | | | | | | NB_TRANS[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | EPT_MAPD | | | | | | | |
| 0x0284 | UDPHS_EPTCTLENB12 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0284 | UDPHS_EPTCTLENB12 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0288 | UDPHS_EPTCTLDIS12 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0288 | UDPHS_EPTCTLDIS12 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x028C | UDPHS_EPTCTL12 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x028C | UDPHS_EPTCTL12 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x0290 ... 0x0293 | Reserved | | | | | | | | | |
| 0x0294 | UDPHS_EPTSETSTA12 | 7:0 | | | FRCESTALL | | | | | |
| | | 15:8 | | | | | TXRDY | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0294 | UDPHS_EPTSETSTA12 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | TXRDY_TRER | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0298 | UDPHS_EPTCLRSTA12 | 7:0 | | TOGGLESQ | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0298 | UDPHS_EPTCLRSTA12 | 7:0 | | TOGGLESQ | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x029C | UDPHS_EPTSTA12 | 7:0 | TOGGLESQ_STA[1:0] | | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK_CTLDIR[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x029C | UDPHS_EPTSTA12 | 7:0 | TOGGLESQ_STA[1:0] | | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x02A0 | UDPHS_EPTCFG13 | 7:0 | BK_NUMBER[1:0] | | EPT_TYPE[1:0] | | EPT_DIR | EPT_SIZE[2:0] | | |
| | | 15:8 | | | | | | | NB_TRANS[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | EPT_MAPD | | | | | | | |
| 0x02A4 | UDPHS_EPTCTLENB13 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x02A4 | UDPHS_EPTCTLENB13 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x02A8 | UDPHS_EPTCTLDIS13 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x02A8 | UDPHS_EPTCTLDIS13 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x02AC | UDPHS_EPTCTL13 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x02AC | UDPHS_EPTCTL13 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x02B0 ... 0x02B3 | Reserved | | | | | | | | | |
| 0x02B4 | UDPHS_EPTSETSTA13 | 7:0 | | | FRCESTALL | | | | | |
| | | 15:8 | | | | | TXRDY | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02B4 | UDPHS_EPTSETSTA13 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | TXRDY_TRER | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02B8 | UDPHS_EPTCLRSTA13 | 7:0 | | TOGGLESQ | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02B8 | UDPHS_EPTCLRSTA13 | 7:0 | | TOGGLESQ | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02BC | UDPHS_EPTSTA13 | 7:0 | TOGGLESQ_STA[1:0] | | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK_CTLDIR[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x02BC | UDPHS_EPTSTA13 | 7:0 | TOGGLESQ_STA[1:0] | | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x02C0 | UDPHS_EPTCFG14 | 7:0 | BK_NUMBER[1:0] | | EPT_TYPE[1:0] | | EPT_DIR | EPT_SIZE[2:0] | | |
| | | 15:8 | | | | | | | NB_TRANS[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | EPT_MAPD | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x02C4 | UDPHS_EPTCTLENB14 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x02C4 | UDPHS_EPTCTLENB14 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x02C8 | UDPHS_EPTCTLDIS14 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x02C8 | UDPHS_EPTCTLDIS14 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x02CC | UDPHS_EPTCTL14 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x02CC | UDPHS_EPTCTL14 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x02D0 ... 0x02D3 | Reserved | | | | | | | | | |
| 0x02D4 | UDPHS_EPTSETSTA14 | 7:0 | | | FRCESTALL | | | | | |
| | | 15:8 | | | | | TXRDY | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02D4 | UDPHS_EPTSETSTA14 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | TXRDY_TRER | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x02D8 | UDPHS_EPTCLRSTA14 | 7:0 | | TOGGLESQ | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02D8 | UDPHS_EPTCLRSTA14 | 7:0 | | TOGGLESQ | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02DC | UDPHS_EPTSTA14 | 7:0 | TOGGLESQ_STA[1:0] | | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK_CTLDIR[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x02DC | UDPHS_EPTSTA14 | 7:0 | TOGGLESQ_STA[1:0] | | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x02E0 | UDPHS_EPTCFG15 | 7:0 | BK_NUMBER[1:0] | | EPT_TYPE[1:0] | | EPT_DIR | EPT_SIZE[2:0] | | |
| | | 15:8 | | | | | | | NB_TRANS[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | EPT_MAPD | | | | | | | |
| 0x02E4 | UDPHS_EPTCTLENB15 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x02E4 | UDPHS_EPTCTLENB15 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x02E8 | UDPHS_EPTCTLDIS15 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x02E8 | UDPHS_EPTCTLDIS15 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x02EC | UDPHS_EPTCTL15 | 7:0 | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x02EC | UDPHS_EPTCTL15 | 7:0 | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | | | | | | BUSY_BANK | | |
| | | 31:24 | SHRT_PCKT | | | | | | | |
| 0x02F0 ... 0x02F3 | Reserved | | | | | | | | | |
| 0x02F4 | UDPHS_EPTSETSTA15 | 7:0 | | | FRCESTALL | | | | | |
| | | 15:8 | | | | | TXRDY | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02F4 | UDPHS_EPTSETSTA15 | 7:0 | | | | | | | | |
| | | 15:8 | | | | | TXRDY_TRER | | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02F8 | UDPHS_EPTCLRSTA15 | 7:0 | | TOGGLESQ | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02F8 | UDPHS_EPTCLRSTA15 | 7:0 | | TOGGLESQ | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | | TX_COMPLT | RXRDY_TXKL | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02FC | UDPHS_EPTSTA15 | 7:0 | TOGGLESQ_STA[1:0] | | FRCESTALL | | | | | |
| | | 15:8 | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK_CTLDIR[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| 0x02FC | UDPHS_EPTSTA15 | 7:0 | TOGGLESQ_STA[1:0] | | | | | | | |
| | | 15:8 | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| | | 23:16 | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK[1:0] | |
| | | 31:24 | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0300 | UDPHS_DMANXTDSC0 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0304 | UDPHS_DMAADDRESS0 | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0308 | UDPHS_DMACONTROL0 | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x030C | UDPHS_DMASTATUS0 | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0310 | UDPHS_DMANXTDSC1 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0314 | UDPHS_DMAADDRESS1 | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0318 | UDPHS_DMACONTROL1 | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x031C | UDPHS_DMASTATUS1 | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0320 | UDPHS_DMANXTDSC2 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0324 | UDPHS_DMAADDRESS2 | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0328 | UDPHS_DMACONTROL2 | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x032C | UDPHS_DMASTATUS2 | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0330 | UDPHS_DMANXTDSC3 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0334 | UDPHS_DMAADDRESS3 | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0338 | UDPHS_DMACONTROL3 | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x033C | UDPHS_DMASTATUS3 | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0340 | UDPHS_DMANXTDSC4 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0344 | UDPHS_DMAADDRESS4 | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0348 | UDPHS_DMACONTROL4 | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x034C | UDPHS_DMASTATUS4 | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0350 | UDPHS_DMANXTDSC5 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0354 | UDPHS_DMAADDRESS5 | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0358 | UDPHS_DMACONTROL5 | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x035C | UDPHS_DMASTATUS5 | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0360 | UDPHS_DMANXTDSC6 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0364 | UDPHS_DMAADDRESS6 | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0368 | UDPHS_DMACONTROL6 | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x036C | UDPHS_DMASTATUS6 | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |

### 40.7.1 UDPHS Control Register

**Name:** UDPHS_CTRL
**Offset:** 0x00
**Reset:** 0x00000200
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PULLD_DIS | REWAKEUP | DETACH | EN_UDPHS |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 1 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FADDR_EN | DEV_ADDR[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 11 – PULLD_DIS** Pulldown Disable (cleared upon USB reset)
When set, there is no pulldown on DP & DM. (DM Pulldown = DP Pulldown = 0).

**Note:** If the DETACH bit is also set, device DP & DM are left in high impedance state.

(See description of bit “DETACH”.)

| DETACH | PULLD_DIS | DP | DM | Condition |
|---|---|---|---|---|
| 0 | 0 | Pullup | Pulldown | Not recommended |
| 0 | 1 | Pullup | High impedance state | VBUS present |
| 1 | 0 | Pulldown | Pulldown | No VBUS |
| 1 | 1 | High impedance state | High impedance state | VBUS present & software disconnect |

**Bit 10 – REWAKEUP** Send Remote Wakeup (cleared upon USB reset)
An Upstream Resume is sent only after the UDPHS bus has been in SUSPEND state for at least 5 ms.

This bit is automatically cleared by hardware at the end of the Upstream Resume.

| Value | Description |
|---|---|
| 0 | Remote Wakeup is disabled (read), or this bit has no effect (write). |

| Value | Description |
|---|---|
| 1 | Remote Wakeup is enabled (read), or this bit forces an external interrupt on the UDPHS controller for Remote Wakeup purposes. |

**Bit 9 – DETACH** Detach Command
See description of bit “PULL_DIS”.

| Value | Description |
|---|---|
| 0 | UDPHS is attached (read), or this bit pulls up the DP line (attach command) (write). |
| 1 | UDPHS is detached, UTMI transceiver is suspended (read), or this bit simulates a detach on the UDPHS line and forces the UTMI transceiver into suspend state (Suspend M = 0) (write). |

**Bit 8 – EN_UDPHS** UDPHS Enable

| Value | Description |
|---|---|
| 0 | UDPHS is disabled (read), or this bit disables and resets the UDPHS controller (write). Switch the host to UTMI. |
| 1 | UDPHS is enabled (read), or this bit enables the UDPHS controller (write). Switch the host to UTMI. |

**Bit 7 – FADDR_EN** Function Address Enable (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Device is not in address state (read), or only the default function address is used (write). |
| 1 | Device is in address state (read), or this bit is set by the device firmware after a successful status phase of a SET_ADDRESS transaction (write). When set, the only address accepted by the UDPHS controller is the one stored in the UDPHS Address field. It will not be cleared afterwards by the device firmware. It is cleared by hardware on hardware reset, or when UDPHS bus reset is received. |

**Bits 6:0 – DEV_ADDR[6:0]** UDPHS Address (cleared upon USB reset)
This field contains the default address (0) after power-up or UDPHS bus reset (read), or it is written with the value set by a SET_ADDRESS request received by the device firmware (write).

### 40.7.2 UDPHS Frame Number Register

**Name:** UDPHS_FNUM
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FNUM_ERR | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | FRAME_NUMBER[10:5] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FRAME_NUMBER[4:0] | | | | | MICRO_FRAME_NUM[2:0] | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – FNUM_ERR** Frame Number CRC Error (cleared upon USB reset)
This bit is set by hardware when a corrupted Frame Number in Start of Frame packet (or Micro SOF) is received.

This bit and the INT_SOF (or MICRO_SOF) interrupt are updated at the same time.

**Bits 13:3 – FRAME_NUMBER[10:0]** Frame Number as defined in the Packet Field Formats (cleared upon USB reset)
This field is provided in the last received SOF packet (see INT_SOF in the UDPHS Interrupt Status Register).

**Bits 2:0 – MICRO_FRAME_NUM[2:0]** Microframe Number (cleared upon USB reset)
Number of the received microframe (0 to 7) in one frame.This field is reset at the beginning of each new frame (1 ms).

One microframe is received each 125 microseconds (1 ms/8).

### 40.7.3 UDPHS Interrupt Enable Register

**Name:** UDPHS_IEN
**Offset:** 0x10
**Reset:** 0x00000010
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DMA_7 | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | EPT_15 | EPT_14 | EPT_13 | EPT_12 | EPT_11 | EPT_10 | EPT_9 | EPT_8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EPT_7 | EPT_6 | EPT_5 | EPT_4 | EPT_3 | EPT_2 | EPT_1 | EPT_0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UPSTR_RES | ENDOFRSM | WAKE_UP | ENDRESET | INT_SOF | MICRO_SOF | DET_SUSPD | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |

**Bits 25, 26, 27, 28, 29, 30, 31 – DMA_x** DMA Channel x Interrupt Enable (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Disable the interrupts for this channel. |
| 1 | Enable the interrupts for this channel. |

**Bits 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 – EPT_x** Endpoint x Interrupt Enable (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Disable the interrupts for this endpoint. |
| 1 | Enable the interrupts for this endpoint. |

**Bit 7 – UPSTR_RES** Upstream Resume Interrupt Enable (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Disable Upstream Resume Interrupt. |
| 1 | Enable Upstream Resume Interrupt. |

**Bit 6 – ENDOFRSM** End Of Resume Interrupt Enable (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Disable Resume Interrupt. |

| Value | Description |
|---|---|
| 1 | Enable Resume Interrupt. |

**Bit 5 – WAKE_UP** Wake Up CPU Interrupt Enable (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Disable Wake-up CPU Interrupt. |
| 1 | Enable Wake-up CPU Interrupt. |

**Bit 4 – ENDRESET** End Of Reset Interrupt Enable (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Disable End Of Reset Interrupt. |
| 1 | Enable End Of Reset Interrupt. Automatically enabled after USB reset. |

**Bit 3 – INT_SOF** SOF Interrupt Enable (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Disable SOF Interrupt. |
| 1 | Enable SOF Interrupt. |

**Bit 2 – MICRO_SOF** Micro-SOF Interrupt Enable (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Disable Micro-SOF Interrupt. |
| 1 | Enable Micro-SOF Interrupt. |

**Bit 1 – DET_SUSPD** Suspend Interrupt Enable (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Disable Suspend Interrupt. |
| 1 | Enable Suspend Interrupt. |

### 40.7.4 UDPHS Interrupt Status Register

**Name:** UDPHS_INTSTA
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DMA_7 | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | |
| Access | R | R | R | R | R | R | R | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | EPT_15 | EPT_14 | EPT_13 | EPT_12 | EPT_11 | EPT_10 | EPT_9 | EPT_8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EPT_7 | EPT_6 | EPT_5 | EPT_4 | EPT_3 | EPT_2 | EPT_1 | EPT_0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UPSTR_RES | ENDOFRSM | WAKE_UP | ENDRESET | INT_SOF | MICRO_SOF | DET_SUSPD | SPEED |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 25, 26, 27, 28, 29, 30, 31 – DMA_x** DMA Channel x Interrupt

| Value | Description |
|---|---|
| 0 | Reset when the UDPHS_DMASTATUSx interrupt source is cleared. |
| 1 | Set by hardware when an interrupt is triggered by the DMA Channelx and this endpoint interrupt is enabled by the DMA_x bit in UDPHS_IEN. |

**Bits 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 – EPT_x** Endpoint x Interrupt (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Reset when the UDPHS_EPTSTAx interrupt source is cleared. |
| 1 | Set by hardware when an interrupt is triggered by the UDPHS_EPTSTAx register and this endpoint interrupt is enabled by the EPT_x bit in UDPHS_IEN. |

**Bit 7 – UPSTR_RES** Upstream Resume Interrupt

| Value | Description |
|---|---|
| 0 | Cleared by setting the UPSTR_RES bit in UDPHS_CLRINT. |
| 1 | Set by hardware when the UDPHS controller is sending a resume signal called “upstream resume”. This triggers a UDPHS interrupt when the UPSTR_RES bit is set in UDPHS_IEN. |

**Bit 6 – ENDOFRSM** End Of Resume Interrupt

| Value | Description |
|---|---|
| 0 | Cleared by setting the ENDOFRSM bit in UDPHS_CLRINT. |
| 1 | Set by hardware when the UDPHS controller detects a good end of resume signal initiated by the host. This triggers a UDPHS interrupt when the ENDOFRSM bit is set in UDPHS_IEN. |

**Bit 5 – WAKE_UP** Wake Up CPU Interrupt

| Value | Description |
|---|---|
| 0 | Cleared by setting the WAKE_UP bit in UDPHS_CLRINT. |
| 1 | Set by hardware when the UDPHS controller is in SUSPEND state and is re-activated by a filtered non-idle signal from the UDPHS line (not by an upstream resume). This triggers a UDPHS interrupt when the WAKE_UP bit is set in UDPHS_IEN register. When receiving this interrupt, the user has to enable the device controller clock prior to operation.<br>**Note:** this interrupt is generated even if the device controller clock is disabled. |

**Bit 4 – ENDRESET** End Of Reset Interrupt

| Value | Description |
|---|---|
| 0 | Cleared by setting the ENDRESET bit in UDPHS_CLRINT. |
| 1 | Set by hardware when an End Of Reset has been detected by the UDPHS controller. This triggers a UDPHS interrupt when the ENDRESET bit is set in UDPHS_IEN. |

**Bit 3 – INT_SOF** Start Of Frame Interrupt
**Note:** The Micro Start Of Frame Interrupt (MICRO_SOF), and the Start Of Frame Interrupt (INT_SOF) are not generated at the same time.

| Value | Description |
|---|---|
| 0 | Cleared by setting the INT_SOF bit in UDPHS_CLRINT. |
| 1 | Set by hardware when an UDPHS Start Of Frame PID (SOF) has been detected (every 1 ms) or synthesized by the macro. This triggers a UDPHS interrupt when the INT_SOF bit is set in UDPHS_IEN register. In case of detected SOF, in High Speed mode, the MICRO_FRAME_NUMBER field is cleared in UDPHS_FNUM register and the FRAME_NUMBER field is updated. |

**Bit 2 – MICRO_SOF** Micro Start Of Frame Interrupt
**Note:** The Micro Start Of Frame Interrupt (MICRO_SOF), and the Start Of Frame Interrupt (INT_SOF) are not generated at the same time.

| Value | Description |
|---|---|
| 0 | Cleared by setting the MICRO_SOF bit in UDPHS_CLRINT register. |
| 1 | Set by hardware when an UDPHS micro start of frame PID (SOF) has been detected (every 125 us) or synthesized by the macro. This triggers a UDPHS interrupt when the MICRO_SOF bit is set in UDPHS_IEN. In case of detected SOF, the MICRO_FRAME_NUM field in UDPHS_FNUM register is incremented and the FRAME_NUMBER field does not change. |

**Bit 1 – DET_SUSPD** Suspend Interrupt

| Value | Description |
|---|---|
| 0 | Cleared by setting the DET_SUSPD bit in UDPHS_CLRINT register. |
| 1 | Set by hardware when a UDPHS Suspend (Idle bus for three frame periods, a J state for 3 ms) is detected. This triggers a UDPHS interrupt when the DET_SUSPD bit is set in UDPHS_IEN register. |

**Bit 0 – SPEED** Speed Status

| Value | Description |
|---|---|
| 0 | Reset by hardware when the hardware is in Full Speed mode. |
| 1 | Set by hardware when the hardware is in High Speed mode. |

### 40.7.5 UDPHS Clear Interrupt Register

**Name:** UDPHS_CLRINT
**Offset:** 0x18
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UPSTR_RES | ENDOFRSM | WAKE_UP | ENDRESET | INT_SOF | MICRO_SOF | DET_SUSPD | |
| Access | W | W | W | W | W | W | W | |
| Reset | – | – | – | – | – | – | – | |

**Bit 7 – UPSTR_RES** Upstream Resume Interrupt Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the UPSTR_RES bit in UDPHS_INTSTA. |

**Bit 6 – ENDOFRSM** End Of Resume Interrupt Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the ENDOFRSM bit in UDPHS_INTSTA. |

**Bit 5 – WAKE_UP** Wake Up CPU Interrupt Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the WAKE_UP bit in UDPHS_INTSTA. |

**Bit 4 – ENDRESET** End Of Reset Interrupt Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the ENDRESET bit in UDPHS_INTSTA. |

**Bit 3 – INT_SOF** Start Of Frame Interrupt Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the INT_SOF bit in UDPHS_INTSTA. |

**Bit 2 – MICRO_SOF** Micro Start Of Frame Interrupt Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the MICRO_SOF bit in UDPHS_INTSTA. |

**Bit 1 – DET_SUSPD** Suspend Interrupt Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the DET_SUSPD bit in UDPHS_INTSTA. |

### 40.7.6 UDPHS Endpoints Reset Register

**Name:** UDPHS_EPTRST
**Offset:** 0x1C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EPT_15 | EPT_14 | EPT_13 | EPT_12 | EPT_11 | EPT_10 | EPT_9 | EPT_8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EPT_7 | EPT_6 | EPT_5 | EPT_4 | EPT_3 | EPT_2 | EPT_1 | EPT_0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – EPT_x** Endpoint x Reset
Setting this bit clears all bits in the Endpoint status UDPHS_EPTSTAx register except the TOGGLESQ_STA field.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Reset the Endpointx state. |

### 40.7.7 UDPHS Test Register

**Name:** UDPHS_TST
**Offset:** 0xE0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | OPMODE2 | TST_PKT | TST_K | TST_J | SPEED_CFG[1:0] | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 5 – OPMODE2** OpMode2
**Note:** For the Test mode, Test_SE0_NAK (refer to Universal Serial Bus Specification, Revision 2.0: 7.1.20, Test Mode Support). Force the device in High Speed mode, and configure a bulk-type endpoint. Do not fill this endpoint for sending NAK to the host.
Upon command, a port's transceiver must enter the High Speed Receive mode and remain in that mode until the exit action is taken. This enables the testing of output impedance, low level output voltage and loading characteristics. In addition, while in this mode, upstream facing ports (and only upstream facing ports) must respond to any IN token packet with a NAK handshake (only if the packet CRC is determined to be correct) within the normal allowed device response time. This enables testing of the device squelch level circuitry and, additionally, provides a general purpose stimulus/response test for basic functional testing.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Set to force the OpMode signal (UTMI interface) to "10", to disable the bit-stuffing and the NRZI encoding. |

**Bit 4 – TST_PKT** Test Packet Mode

| Value | Description |
|---|---|
| 0 | No effect. |

| Value | Description |
|---|---|
| 1 | Set to repetitively transmit the packet stored in the current bank. This enables the testing of rise and fall times, eye patterns, jitter, and any other dynamic waveform specifications. |

**Bit 3 – TST_K** Test K Mode

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Set to send the K state on the UDPHS line. This enables the testing of the high output drive level on the D- line. |

**Bit 2 – TST_J** Test J Mode

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Set to send the J state on the UDPHS line. This enables the testing of the high output drive level on the D+ line. |

**Bits 1:0 – SPEED_CFG[1:0]** Speed Configuration

| Value | Name | Description |
|---|---|---|
| 0 | NORMAL | Normal mode: The macro is in Full Speed mode, ready to make a High Speed identification, if the host supports it and then to automatically switch to High Speed mode. |
| 1 | – | Reserved |
| 2 | HIGH_SPEED | Force High Speed: Set this value to force the hardware to work in High Speed mode. Only for debug or test purpose. |
| 3 | FULL_SPEED | Force Full Speed: Set this value to force the hardware to work only in Full Speed mode. In this configuration, the macro will not respond to a High Speed reset handshake. |

### 40.7.8 UDPHS Endpoint Configuration Register

**Name:** UDPHS_EPTCFGx
**Offset:** 0x0100 + x*0x20 [x=0..15]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | EPT_MAPD | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | NB_TRANS[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BK_NUMBER[1:0] | | EPT_TYPE[1:0] | | EPT_DIR | EPT_SIZE[2:0] | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – EPT_MAPD** Endpoint Mapped (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | The user should reprogram the register with correct values. |
| 1 | Set by hardware when the endpoint size (EPT_SIZE) and the number of banks (BK_NUMBER) are correct regarding:<br>– The max endpoint size for this endpoint<br>– The number of allowed banks for this endpoint<br>– The number of endpoints/banks already allocated<br>– The FIFO max capacity (FIFO_MAX_SIZE in UDPHS_IPFEATURES register) |

**Bits 9:8 – NB_TRANS[1:0]** Number Of Transactions per Microframe (cleared upon USB reset)
The number of transactions per microframe is set by software.

**Note:** Meaningful for high bandwidth isochronous endpoint only.

**Bits 7:6 – BK_NUMBER[1:0]** Number of Banks (cleared upon USB reset)
Set this field according to the endpoint's number of banks (see section Endpoint Configuration).

| Value | Name | Description |
|---|---|---|
| 0 | 0 | Zero bank, the endpoint is not mapped in memory |

| Value | Name | Description |
|---|---|---|
| 1 | 1 | One bank (bank 0) |
| 2 | 2 | Double bank (Ping-Pong: bank0/bank1) |
| 3 | 3 | Triple bank (bank0/bank1/bank2) |

**Bits 5:4 – EPT_TYPE[1:0]** Endpoint Type (cleared upon USB reset)
Set this field according to the endpoint type (see section Endpoint Configuration).

(Endpoint 0 should always be configured as control).

| Value | Name | Description |
|---|---|---|
| 0 | CTRL8 | Control endpoint |
| 1 | ISO | Isochronous endpoint |
| 2 | BULK | Bulk endpoint |
| 3 | INT | Interrupt endpoint |

**Bit 3 – EPT_DIR** Endpoint Direction (cleared upon USB reset)
For Control endpoints this bit has no effect and should be left at zero.

| Value | Description |
|---|---|
| 0 | Clear this bit to configure OUT direction for Bulk, Interrupt and Isochronous endpoints. |
| 1 | Set this bit to configure IN direction for Bulk, Interrupt and Isochronous endpoints. |

**Bits 2:0 – EPT_SIZE[2:0]** Endpoint Size (cleared upon USB reset)
Set this field according to the endpoint size in bytes (see the section Endpoint Configuration). Note that 1024 bytes is only for isochronous endpoints.

| Value | Name | Description |
|---|---|---|
| 0 | 8 | 8 bytes |
| 1 | 16 | 16 bytes |
| 2 | 32 | 32 bytes |
| 3 | 64 | 64 bytes |
| 4 | 128 | 128 bytes |
| 5 | 256 | 256 bytes |
| 6 | 512 | 512 bytes |
| 7 | 1024 | 1024 bytes |

© 2018 Microchip Technology Inc.

### 40.7.9 UDPHS Endpoint Control Enable Register (Control, Bulk, Interrupt Endpoints)

**Name:** UDPHS_EPTCTLENBx
**Offset:** 0x0104 + x*0x20 [x=0..15]
**Reset:** –
**Property:** Write-only

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in UDPHS Endpoint Configuration Register.

For additional information, see UDPHS Endpoint Control Register (Control, Bulk, Interrupt Endpoints).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SHRT_PCKT | | | | | | | |
| Access | W | | | | | | | |
| Reset | – | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | BUSY_BANK | | |
| Access | | | | | | W | | |
| Reset | | | | | | – | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| Access | | | | W | W | | W | W |
| Reset | | | | – | – | | – | – |

**Bit 31 – SHRT_PCKT** Short Packet Send/Short Packet Interrupt Enable
For IN endpoints: Guarantees short packet at end of DMA Transfer if the UDPHS_DMACONTROLx register END_B_EN and UDPHS_EPTCTLx register AUTOVALID bits are also set.

For OUT endpoints:

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable Short Packet Interrupt. |

**Bit 18 – BUSY_BANK** Busy Bank Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable Busy Bank Interrupt. |

**Bit 15 – NAK_OUT** NAKOUT Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable NAKOUT Interrupt. |

**Bit 14 – NAK_IN** NAKIN Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable NAKIN Interrupt. |

**Bit 13 – STALL_SNT** Stall Sent Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable Stall Sent Interrupt. |

**Bit 12 – RX_SETUP** Received SETUP

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable RX_SETUP Interrupt. |

**Bit 11 – TXRDY** TX Packet Ready Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable TX Packet Ready/Transaction Error Interrupt. |

**Bit 10 – TX_COMPLT** Transmitted IN Data Complete Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable Transmitted IN Data Complete Interrupt. |

**Bit 9 – RXRDY_TXKL** Received OUT Data Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable Received OUT Data Interrupt. |

**Bit 8 – ERR_OVFLW** Overflow Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable Overflow Error Interrupt. |

**Bit 4 – NYET_DIS** NYET Disable (Only for High Speed Bulk OUT endpoints)

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Forces an ACK response to the next High Speed Bulk OUT transfer instead of a NYET response. |

**Bit 3 – INTDIS_DMA** Interrupts Disable DMA

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If set, when an enabled endpoint-originated interrupt is triggered, the DMA request is disabled. |

**Bit 1 – AUTO_VALID** Packet Auto-Valid Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable this bit to automatically validate the current packet and switch to the next bank for both IN and OUT transfers. |

**Bit 0 – EPT_ENABL** Endpoint Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable endpoint according to the device configuration. |

### 40.7.10 UDPHS Endpoint Control Enable Register (Isochronous Endpoints)

**Name:** UDPHS_EPTCTLENBx
**Offset:** 0x0104 + x*0x20 [x=0..15]
**Reset:** –
**Property:** Write-only

This register view is relevant only if EPT_TYPE = 0x1 in UDPHS Endpoint Configuration Register.

For additional information, see UDPHS Endpoint Control Register (Isochronous Endpoint).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SHRT_PCKT | | | | | | | |
| Access | W | | | | | | | |
| Reset | – | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | BUSY_BANK | | |
| Access | | | | | | W | | |
| Reset | | | | | | – | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| Access | | W | W | W | W | W | W | W |
| Reset | | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| Access | W | W | | | W | | W | W |
| Reset | – | – | | | – | | – | – |

**Bit 31 – SHRT_PCKT** Short Packet Send/Short Packet Interrupt Enable
For IN endpoints: Guarantees short packet at end of DMA Transfer if the UDPHS_DMACONTROLx register END_B_EN and UDPHS_EPTCTLx register AUTOVALID bits are also set.

For OUT endpoints:

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable Short Packet Interrupt. |

**Bit 18 – BUSY_BANK** Busy Bank Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable Busy Bank Interrupt. |

**Bit 14 – ERR_FLUSH** Bank Flush Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable Bank Flush Error Interrupt. |

**Bit 13 – ERR_CRC_NTR** ISO CRC Error/Number of Transaction Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable Error CRC ISO/Error Number of Transaction Interrupt. |

**Bit 12 – ERR_FL_ISO** Error Flow Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable Error Flow ISO Interrupt. |

**Bit 11 – TXRDY_TRER** TX Packet Ready/Transaction Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable TX Packet Ready/Transaction Error Interrupt. |

**Bit 10 – TX_COMPLT** Transmitted IN Data Complete Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable Transmitted IN Data Complete Interrupt. |

**Bit 9 – RXRDY_TXKL** Received OUT Data Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable Received OUT Data Interrupt. |

**Bit 8 – ERR_OVFLW** Overflow Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable Overflow Error Interrupt. |

**Bit 7 – MDATA_RX** MDATA Interrupt Enable (Only for high bandwidth Isochronous OUT endpoints)

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable MDATA Interrupt. |

**Bit 6 – DATAX_RX** DATAx Interrupt Enable (Only for high bandwidth Isochronous OUT endpoints)

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable DATAx Interrupt. |

**Bit 3 – INTDIS_DMA** Interrupts Disable DMA

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If set, when an enabled endpoint-originated interrupt is triggered, the DMA request is disabled. |

**Bit 1 – AUTO_VALID** Packet Auto-Valid Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable this bit to automatically validate the current packet and switch to the next bank for both IN and OUT transfers. |

**Bit 0 – EPT_ENABL** Endpoint Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable endpoint according to the device configuration. |

### 40.7.11 UDPHS Endpoint Control Disable Register (Control, Bulk, Interrupt Endpoints)

**Name:** UDPHS_EPTCTLDISx
**Offset:** 0x0108 + x*0x20 [x=0..15]
**Reset:** –
**Property:** Write-only

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in UDPHS Endpoint Configuration Register.

For additional information, see UDPHS Endpoint Control Register (Control, Bulk, Interrupt Endpoints).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SHRT_PCKT | | | | | | | |
| Access | W | | | | | | | |
| Reset | – | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | BUSY_BANK | | |
| Access | | | | | | W | | |
| Reset | | | | | | – | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| Access | | | | W | W | | W | W |
| Reset | | | | – | – | | – | – |

**Bit 31 – SHRT_PCKT** Short Packet Interrupt Disable
For IN endpoints: Never automatically add a zero length packet at end of DMA transfer.

For OUT endpoints:

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable Short Packet Interrupt. |

**Bit 18 – BUSY_BANK** Busy Bank Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable Busy Bank Interrupt. |

**Bit 15 – NAK_OUT** NAKOUT Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |

| Value | Description |
|---|---|
| 1 | Disable NAKOUT Interrupt. |

**Bit 14 – NAK_IN** NAKIN Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable NAKIN Interrupt. |

**Bit 13 – STALL_SNT** Stall Sent Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable Stall Sent Interrupt. |

**Bit 12 – RX_SETUP** Received SETUP Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable RX_SETUP Interrupt. |

**Bit 11 – TXRDY** TX Packet Ready Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable TX Packet Ready/Transaction Error Interrupt. |

**Bit 10 – TX_COMPLT** Transmitted IN Data Complete Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable Transmitted IN Data Complete Interrupt. |

**Bit 9 – RXRDY_TXKL** Received OUT Data Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable Received OUT Data Interrupt. |

**Bit 8 – ERR_OVFLW** Overflow Error Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable Overflow Error Interrupt. |

**Bit 4 – NYET_DIS** NYET Enable (Only for High Speed Bulk OUT endpoints)

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Let the hardware handle the handshake response for the High Speed Bulk OUT transfer. |

**Bit 3 – INTDIS_DMA** Interrupts Disable DMA

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable the “Interrupts Disable DMA”. |

**Bit 1 – AUTO_VALID** Packet Auto-Valid Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable this bit to not automatically validate the current packet. |

**Bit 0 – EPT_DISABL** Endpoint Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable endpoint. |

### 40.7.12 UDPHS Endpoint Control Disable Register (Isochronous Endpoint)

**Name:** UDPHS_EPTCTLDISx
**Offset:** 0x0108 + x*0x20 [x=0..15]
**Reset:** –
**Property:** Write-only

This register view is relevant only if EPT_TYPE = 0x1 in UDPHS Endpoint Configuration Register.

For additional information, see “UDPHS Endpoint Control Register (Isochronous Endpoint)”.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SHRT_PCKT | | | | | | | |
| Access | W | | | | | | | |
| Reset | – | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | BUSY_BANK | | |
| Access | | | | | | W | | |
| Reset | | | | | | – | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| Access | | W | W | W | W | W | W | W |
| Reset | | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_DISABL |
| Access | W | W | | | W | | W | W |
| Reset | – | – | | | – | | – | – |

**Bit 31 – SHRT_PCKT** Short Packet Interrupt Disable
For IN endpoints: Never automatically add a zero length packet at end of DMA transfer.

For OUT endpoints:

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable Short Packet Interrupt. |

**Bit 18 – BUSY_BANK** Busy Bank Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable Busy Bank Interrupt. |

**Bit 14 – ERR_FLUSH** bank flush error Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |

| Value | Description |
|---|---|
| 1 | Disable Bank Flush Error Interrupt. |

**Bit 13 – ERR_CRC_NTR** ISO CRC Error/Number of Transaction Error Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable Error CRC ISO/Error Number of Transaction Interrupt. |

**Bit 12 – ERR_FL_ISO** Error Flow Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable Error Flow ISO Interrupt. |

**Bit 11 – TXRDY_TRER** TX Packet Ready/Transaction Error Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable TX Packet Ready/Transaction Error Interrupt. |

**Bit 10 – TX_COMPLT** Transmitted IN Data Complete Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable Transmitted IN Data Complete Interrupt. |

**Bit 9 – RXRDY_TXKL** Received OUT Data Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable Received OUT Data Interrupt. |

**Bit 8 – ERR_OVFLW** Overflow Error Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable Overflow Error Interrupt. |

**Bit 7 – MDATA_RX** MDATA Interrupt Disable (Only for High Bandwidth Isochronous OUT endpoints)

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable MDATA Interrupt. |

**Bit 6 – DATAX_RX** DATAx Interrupt Disable (Only for High Bandwidth Isochronous OUT endpoints)

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable DATAx Interrupt. |

**Bit 3 – INTDIS_DMA** Interrupts Disable DMA

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable the “Interrupts Disable DMA”. |

**Bit 1 – AUTO_VALID** Packet Auto-Valid Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable this bit to not automatically validate the current packet. |

**Bit 0 – EPT_DISABL** Endpoint Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable endpoint. |

### 40.7.13 UDPHS Endpoint Control Register (Control, Bulk, Interrupt Endpoints)

**Name:** UDPHS_EPTCTLx
**Offset:** 0x010C + x*0x20 [x=0..15]
**Reset:** 0x00000000
**Property:** Read-only

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in UDPHS Endpoint Configuration Register.

The reset value for UDPHS_EPTCTL0 is 0x00000001.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SHRT_PCKT | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | BUSY_BANK | | |
| Access | | | | | | R | | |
| Reset | | | | | | 0 | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | NYET_DIS | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| Access | | | | R | R | | R | R |
| Reset | | | | 0 | 0 | | 0 | 0 |

**Bit 31 – SHRT_PCKT** Short Packet Interrupt Enabled (cleared upon USB reset)
For OUT endpoints: sends an Interrupt when a Short Packet has been received.

For IN endpoints: a Short Packet transmission is guaranteed upon end of the DMA Transfer, thus signaling a BULK or INTERRUPT end of transfer, but only if the UDPHS_DMACONTROLx register END_B_EN and UDPHS_EPTCTLx register AUTO_VALID bits are also set.

| Value | Description |
|---|---|
| 0 | Short Packet Interrupt is masked. |
| 1 | Short Packet Interrupt is enabled. |

**Bit 18 – BUSY_BANK** Busy Bank Interrupt Enabled (cleared upon USB reset)
For OUT endpoints: an interrupt is sent when all banks are busy.

For IN endpoints: an interrupt is sent when all banks are free.

| Value | Description |
|---|---|
| 0 | BUSY_BANK Interrupt is masked. |
| 1 | BUSY_BANK Interrupt is enabled. |

**Bit 15 – NAK_OUT** NAKOUT Interrupt Enabled (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | NAKOUT Interrupt is masked. |
| 1 | NAKOUT Interrupt is enabled. |

**Bit 14 – NAK_IN** NAKIN Interrupt Enabled (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | NAKIN Interrupt is masked. |
| 1 | NAKIN Interrupt is enabled. |

**Bit 13 – STALL_SNT** Stall Sent Interrupt Enabled (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Stall Sent Interrupt is masked. |
| 1 | Stall Sent Interrupt is enabled. |

**Bit 12 – RX_SETUP** Received SETUP Interrupt Enabled (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Received SETUP is masked. |
| 1 | Received SETUP is enabled. |

**Bit 11 – TXRDY** TX Packet Ready Interrupt Enabled (cleared upon USB reset)

**⚠ CAUTION** Interrupt source is active as long as the corresponding UDPHS_EPTSTAx register TXRDY flag remains low. If there are no more banks available for transmitting after the software has set UDPHS_EPTSTAx/TXRDY for the last transmit packet, then the interrupt source remains inactive until the first bank becomes free again to transmit at UDPHS_EPTSTAx/TXRDY hardware clear.

| Value | Description |
|---|---|
| 0 | TX Packet Ready Interrupt is masked. |
| 1 | TX Packet Ready Interrupt is enabled. |

**Bit 10 – TX_COMPLT** Transmitted IN Data Complete Interrupt Enabled (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Transmitted IN Data Complete Interrupt is masked. |
| 1 | Transmitted IN Data Complete Interrupt is enabled. |

**Bit 9 – RXRDY_TXKL** Received OUT Data Interrupt Enabled (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Received OUT Data Interrupt is masked. |
| 1 | Received OUT Data Interrupt is enabled. |

**Bit 8 – ERR_OVFLW** Overflow Error Interrupt Enabled (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Overflow Error Interrupt is masked. |
| 1 | Overflow Error Interrupt is enabled. |

**Bit 4 – NYET_DIS** NYET Disable (Only for High Speed Bulk OUT Endpoints) (cleared upon USB reset)
**Note:** According to the Universal Serial Bus Specification, Rev 2.0 (8.5.1.1 NAK Responses to OUT/ DATA During PING Protocol), a NAK response to an HS Bulk OUT transfer is expected to be an unusual occurrence.

| Value | Description |
|---|---|
| 0 | Lets the hardware handle the handshake response for the High Speed Bulk OUT transfer. |
| 1 | Forces an ACK response to the next High Speed Bulk OUT transfer instead of a NYET response. |

**Bit 3 – INTDIS_DMA** Interrupt Disables DMA (cleared upon USB reset)
If set, when an enabled endpoint-originated interrupt is triggered, the DMA request is disabled regardless of the UDPHS_IEN register EPT_x bit for this endpoint. Then, the firmware will have to clear or disable the interrupt source or clear this bit if transfer completion is needed.

If the exception raised is associated with the new system bank packet, then the previous DMA packet transfer is normally completed, but the new DMA packet transfer is not started (not requested).

If the exception raised is not associated to a new system bank packet (NAK_IN, NAK_OUT, etc.), then the request cancellation may happen at any time and may immediately stop the current DMA transfer.

This may be used, for example, to identify or prevent an erroneous packet to be transferred into a buffer or to complete a DMA buffer by software after reception of a short packet.

**Bit 1 – AUTO_VALID** Packet Auto-Valid Enabled (Not for CONTROL Endpoints) (cleared upon USB reset)
Set this bit to automatically validate the current packet and switch to the next bank for both IN and OUT endpoints.

**For IN Transfer:**

If this bit is set, the UDPHS_EPTSTAx register TXRDY bit is set automatically when the current bank is full and at the end of DMA buffer if the UDPHS_DMACONTROLx register END_B_EN bit is set.

The user may still set the UDPHS_EPTSTAx register TXRDY bit if the current bank is not full, unless the user needs to send a Zero Length Packet by software.

**For OUT Transfer:**

If this bit is set, the UDPHS_EPTSTAx register RXRDY_TXKL bit is automatically reset for the current bank when the last packet byte has been read from the bank FIFO or at the end of DMA buffer if the UDPHS_DMACONTROLx register END_B_EN bit is set. For example, to truncate a padded data packet when the actual data transfer size is reached.

The user may still clear the UDPHS_EPTSTAx register RXRDY_TXKL bit, for example, after completing a DMA buffer by software if UDPHS_DMACONTROLx register END_B_EN bit was disabled or in order to cancel the read of the remaining data bank(s).

**Bit 0 – EPT_ENABL** Endpoint Enable (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | The endpoint is disabled according to the device configuration. Endpoint 0 should always be enabled after a hardware or UDPHS bus reset and participate in the device configuration. |
| 1 | The endpoint is enabled according to the device configuration. |

### 40.7.14 UDPHS Endpoint Control Register (Isochronous Endpoint)

**Name:** UDPHS_EPTCTLx
**Offset:** 0x010C + x*0x20 [x=0..15]
**Reset:** 0x00000000
**Property:** Read-only

This register view is relevant only if EPT_TYPE = 0x1 in UDPHS Endpoint Configuration Register.

The reset value for UDPHS_EPTCTL0 is 0x00000001.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SHRT_PCKT | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | BUSY_BANK | | |
| Access | | | | | | R | | |
| Reset | | | | | | 0 | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | ERR_FLUSH | ERR_CRC_NT | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MDATA_RX | DATAX_RX | | | INTDIS_DMA | | AUTO_VALID | EPT_ENABL |
| Access | R | R | | | R | | R | R |
| Reset | 0 | 0 | | | 0 | | 0 | 0 |

**Bit 31 – SHRT_PCKT** Short Packet Interrupt Enabled (cleared upon USB reset)
For OUT endpoints: Send an Interrupt when a Short Packet has been received.

For IN endpoints: A Short Packet transmission is guaranteed upon end of the DMA Transfer, thus signaling an end of isochronous (micro-)frame data, but only if the UDPHS_DMACONTROLx register END_B_EN and UDPHS_EPTCTLx register AUTO_VALID bits are also set.

| Value | Description |
|---|---|
| 0 | Short Packet Interrupt is masked. |
| 1 | Short Packet Interrupt is enabled. |

**Bit 18 – BUSY_BANK** Busy Bank Interrupt Enabled (cleared upon USB reset)
For OUT endpoints: An interrupt is sent when all banks are busy.

For IN endpoints: An interrupt is sent when all banks are free.

| Value | Description |
|---|---|
| 0 | BUSY_BANK Interrupt is masked. |
| 1 | BUSY_BANK Interrupt is enabled. |

**Bit 14 – ERR_FLUSH** Bank Flush Error Interrupt Enabled (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Bank Flush Error Interrupt is masked. |
| 1 | Bank Flush Error Interrupt is enabled. |

**Bit 13 – ERR_CRC_NTR** ISO CRC Error/Number of Transaction Error Interrupt Enabled (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | ISO CRC error/number of Transaction Error Interrupt is masked. |
| 1 | ISO CRC error/number of Transaction Error Interrupt is enabled. |

**Bit 12 – ERR_FL_ISO** Error Flow Interrupt Enabled (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Error Flow Interrupt is masked. |
| 1 | Error Flow Interrupt is enabled. |

**Bit 11 – TXRDY_TRER** TX Packet Ready/Transaction Error Interrupt Enabled (cleared upon USB reset)

⚠ CAUTION Interrupt source is active as long as the corresponding UDPHS_EPTSTAx register TXRDY_TRER flag remains low. If there are no more banks available for transmitting after the software has set UDPHS_EPTSTAx/TXRDY_TRER for the last transmit packet, then the interrupt source remains inactive until the first bank becomes free again to transmit at UDPHS_EPTSTAx/TXRDY_TRER hardware clear.

| Value | Description |
|---|---|
| 0 | TX Packet Ready/Transaction Error Interrupt is masked. |
| 1 | TX Packet Ready/Transaction Error Interrupt is enabled. |

**Bit 10 – TX_COMPLT** Transmitted IN Data Complete Interrupt Enabled (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Transmitted IN Data Complete Interrupt is masked. |
| 1 | Transmitted IN Data Complete Interrupt is enabled. |

**Bit 9 – RXRDY_TXKL** Received OUT Data Interrupt Enabled (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Received OUT Data Interrupt is masked. |
| 1 | Received OUT Data Interrupt is enabled. |

**Bit 8 – ERR_OVFLW** Overflow Error Interrupt Enabled (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | Overflow Error Interrupt is masked. |
| 1 | Overflow Error Interrupt is enabled. |

**Bit 7 – MDATA_RX** MDATA Interrupt Enabled (Only for High Bandwidth Isochronous OUT endpoints) (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Send an interrupt when an MDATA packet has been received and so at least one packet of the microframe data payload has been received. |

**Bit 6 – DATAX_RX** DATAx Interrupt Enabled (Only for High Bandwidth Isochronous OUT endpoints) (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Send an interrupt when a DATA2, DATA1 or DATA0 packet has been received meaning the whole microframe data payload has been received. |

**Bit 3 – INTDIS_DMA** Interrupt Disables DMA (cleared upon USB reset)
If set, when an enabled endpoint-originated interrupt is triggered, the DMA request is disabled regardless of the UDPHS_IEN register EPT_x bit for this endpoint. Then, the firmware will have to clear or disable the interrupt source or clear this bit if transfer completion is needed.

If the exception raised is associated with the new system bank packet, then the previous DMA packet transfer is normally completed, but the new DMA packet transfer is not started (not requested).

If the exception raised is not associated to a new system bank packet (ex: ERR_FL_ISO), then the request cancellation may happen at any time and may immediately stop the current DMA transfer.

This may be used, for example, to identify or prevent an erroneous packet to be transferred into a buffer or to complete a DMA buffer by software after reception of a short packet, or to perform buffer truncation on ERR_FL_ISO interrupt for adaptive rate.

**Bit 1 – AUTO_VALID** Packet Auto-Valid Enabled (cleared upon USB reset)
Set this bit to automatically validate the current packet and switch to the next bank for both IN and OUT endpoints.

**For IN Transfer:**
If this bit is set, the UDPHS_EPTSTAx register TXRDY_TRER bit is set automatically when the current bank is full and at the end of DMA buffer if the UDPHS_DMACONTROLx register END_B_EN bit is set.

The user may still set the UDPHS_EPTSTAx register TXRDY_TRER bit if the current bank is not full, unless the user needs to send a Zero Length Packet by software.

**For OUT Transfer:**
If this bit is set, the UDPHS_EPTSTAx register RXRDY_TXKL bit is automatically reset for the current bank when the last packet byte has been read from the bank FIFO or at the end of DMA buffer if the UDPHS_DMACONTROLx register END_B_EN bit is set. For example, to truncate a padded data packet when the actual data transfer size is reached.

The user may still clear the UDPHS_EPTSTAx register RXRDY_TXKL bit, for example, after completing a DMA buffer by software if UDPHS_DMACONTROLx register END_B_EN bit was disabled or in order to cancel the read of the remaining data bank(s).

**Bit 0 – EPT_ENABL** Endpoint Enable (cleared upon USB reset)

| Value | Description |
|---|---|
| 0 | The endpoint is disabled according to the device configuration. Endpoint 0 should always be enabled after a hardware or UDPHS bus reset and participate in the device configuration. |
| 1 | The endpoint is enabled according to the device configuration. |

### 40.7.15 UDPHS Endpoint Set Status Register (Control, Bulk, Interrupt Endpoints)

**Name:** UDPHS_EPTSETSTAx
**Offset:** 0x0114 + x*0x20 [x=0..15]
**Reset:** –
**Property:** Write-only

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in UDPHS Endpoint Configuration Register.

For additional information, see UDPHS Endpoint Status Register (Control, Bulk, Interrupt Endpoints).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TXRDY | | RXRDY_TXKL | |
| Access | | | | | W | | W | |
| Reset | | | | | – | | – | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | FRCESTALL | | | | | |
| Access | | | W | | | | | |
| Reset | | | – | | | | | |

**Bit 11 – TXRDY** TX Packet Ready Set

| Value | Description |
|---|---|
| 0 | No effect. |

| Value | Description |
|---|---|
| 1 | Set this bit after a packet has been written into the endpoint FIFO for IN data transfers<br>– This flag is used to generate a Data IN transaction (device to host).<br>– Device firmware checks that it can write a data payload in the FIFO, checking that TXRDY is cleared.<br>– Transfer to the FIFO is done by writing in the “Buffer Address” register.<br>– Once the data payload has been transferred to the FIFO, the firmware notifies the UDPHS device setting TXRDY to one.<br>– UDPHS bus transactions can start.<br>– TXCOMP is set once the data payload has been received by the host.<br>– Data should be written into the endpoint FIFO only after this bit has been cleared.<br>– Set this bit without writing data to the endpoint FIFO to send a Zero Length Packet. |

**Bit 9 – RXRDY_TXKL** KILL Bank Set (for IN Endpoint)

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Kill the last written bank. |

**Bit 5 – FRCESTALL** Stall Handshake Request Set
Refer to chapters 8.4.5 (Handshake Packets) and 9.4.5 (Get Status) of the Universal Serial Bus Specification, Rev 2.0 for more information on the STALL handshake.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Set this bit to request a STALL answer to the host for the next handshake |

### 40.7.16 UDPHS Endpoint Set Status Register (Isochronous Endpoint)

**Name:** UDPHS_EPTSETSTAx
**Offset:** 0x0114 + x*0x20 [x=0..15]
**Reset:** –
**Property:** Write-only

This register view is relevant only if EPT_TYPE = 0x1 in UDPHS Endpoint Configuration Register.

For additional information, see UDPHS Endpoint Status Register (Isochronous Endpoint).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TXRDY_TRER | | RXRDY_TXKL | |
| Access | | | | | W | | W | |
| Reset | | | | | – | | – | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 11 – TXRDY_TRER** TX Packet Ready Set

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Set this bit after a packet has been written into the endpoint FIFO for IN data transfers<br>– This flag is used to generate a Data IN transaction (device to host).<br>– Device firmware checks that it can write a data payload in the FIFO, checking that TXRDY_TRER is cleared.<br>– Transfer to the FIFO is done by writing in the “Buffer Address” register.<br>– Once the data payload has been transferred to the FIFO, the firmware notifies the UDPHS device setting TXRDY_TRER to one.<br>– UDPHS bus transactions can start.<br>– TXCOMP is set once the data payload has been sent.<br>– Data should be written into the endpoint FIFO only after this bit has been cleared.<br>– Set this bit without writing data to the endpoint FIFO to send a Zero Length Packet. |

**Bit 9 – RXRDY_TXKL** KILL Bank Set (for IN Endpoint)

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Kill the last written bank. |

### 40.7.17 UDPHS Endpoint Clear Status Register (Control, Bulk, Interrupt Endpoints)

**Name:** UDPHS_EPTCLRSTAx
**Offset:** 0x0118 + x*0x20 [x=0..15]
**Reset:** –
**Property:** Write-only

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in UDPHS Endpoint Configuration Register.

For additional information, see UDPHS Endpoint Status Register (Control, Bulk, Interrupt Endpoints).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | | TX_COMPLT | RXRDY_TXKL | |
| Access | W | W | W | W | | W | W | |
| Reset | – | – | – | – | | – | – | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TOGGLESQ | FRCESTALL | | | | | |
| Access | | W | W | | | | | |
| Reset | | – | – | | | | | |

**Bit 15 – NAK_OUT** NAKOUT Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the NAK_OUT flag of UDPHS_EPTSTAx. |

**Bit 14 – NAK_IN** NAKIN Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the NAK_IN flags of UDPHS_EPTSTAx. |

**Bit 13 – STALL_SNT** Stall Sent Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the STALL_SNT flags of UDPHS_EPTSTAx. |

**Bit 12 – RX_SETUP** Received SETUP Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the RX_SETUP flags of UDPHS_EPTSTAx. |

**Bit 10 – TX_COMPLT** Transmitted IN Data Complete Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the TX_COMPLT flag of UDPHS_EPTSTAx. |

**Bit 9 – RXRDY_TXKL** Received OUT Data Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the RXRDY_TXKL flag of UDPHS_EPTSTAx. |

**Bit 6 – TOGGLESQ** Data Toggle Clear
For OUT endpoints, the next received packet should be a DATA0.

For IN endpoints, the next packet will be sent with a DATA0 PID.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the PID data of the current bank |

**Bit 5 – FRCESTALL** Stall Handshake Request Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the STALL request. The next packets from host will not be STALLed. |

### 40.7.18 UDPHS Endpoint Clear Status Register (Isochronous Endpoint)

**Name:** UDPHS_EPTCLRSTAx
**Offset:** 0x0118 + x*0x20 [x=0..15]
**Reset:** –
**Property:** Write-only

This register view is relevant only if EPT_TYPE = 0x1 in UDPHS Endpoint Configuration Register.

For additional information, see UDPHS Endpoint Status Register (Isochronous Endpoint).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | ERR_FLUSH | ERR_CRC_NTR | ERR_FL_ISO | | TX_COMPLT | RXRDY_TXKL | |
| Access | | W | W | W | | W | W | |
| Reset | | – | – | – | | – | – | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TOGGLESQ | | | | | | |
| Access | | W | | | | | | |
| Reset | | – | | | | | | |

**Bit 14 – ERR_FLUSH** Bank Flush Error Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the ERR_FLUSH flags of UDPHS_EPTSTAx. |

**Bit 13 – ERR_CRC_NTR** Number of Transaction Error Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the ERR_CRC_NTR flags of UDPHS_EPTSTAx. |

**Bit 12 – ERR_FL_ISO** Error Flow Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the ERR_FL_ISO flags of UDPHS_EPTSTAx. |

**Bit 10 – TX_COMPLT** Transmitted IN Data Complete Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the TX_COMPLT flag of UDPHS_EPTSTAx. |

**Bit 9 – RXRDY_TXKL** Received OUT Data Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the RXRDY_TXKL flag of UDPHS_EPTSTAx. |

**Bit 6 – TOGGLESQ** Data Toggle Clear
For OUT endpoints, the next received packet should be a DATA0.

For IN endpoints, the next packet will be sent with a DATA0 PID.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the PID data of the current bank |

### 40.7.19 UDPHS Endpoint Status Register (Control, Bulk, Interrupt Endpoints)

**Name:** UDPHS_EPTSTAx
**Offset:** 0x011C + x*0x20 [x=0..15]
**Reset:** 0x00000040
**Property:** Read-only

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in UDPHS Endpoint Configuration Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK_CTLDIR[1:0] | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NAK_OUT | NAK_IN | STALL_SNT | RX_SETUP | TXRDY | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TOGGLESQ_STA[1:0] | | FRCESTALL | | | | | |
| Access | R | R | R | | | | | |
| Reset | 0 | 1 | 0 | | | | | |

**Bit 31 – SHRT_PCKT** Short Packet (cleared upon USB reset)
An OUT Short Packet is detected when the receive byte count is less than the configured UDPHS_EPTCFGx register EPT_Size.

This bit is updated at the same time as the BYTE_COUNT field.

It is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

**Bits 30:20 – BYTE_COUNT[10:0]** UDPHS Byte Count (cleared upon USB reset)
Byte count of a received data packet.

This field is incremented after each write into the endpoint (to prepare an IN transfer). It is decremented after each reading into the endpoint (OUT transfer).

This field is also updated at RXRDY_TXKL flag clear with the next bank, and at TXRDY flag set with the next bank.

This field is reset by UDPHS_EPTRST.EPT_x.

**Bits 19:18 – BUSY_BANK_STA[1:0]** Busy Bank Number (cleared upon USB reset)
These bits are set by hardware to indicate the number of busy banks.

IN endpoint: Indicates the number of busy banks filled by the user, ready for IN transfer.

OUT endpoint: Indicates the number of busy banks filled by OUT transaction from the Host.

| Value | Name | Description |
|---|---|---|
| 0 | 0BUSYBANK | All banks are free |
| 1 | 1BUSYBANK | 1 busy bank |
| 2 | 2BUSYBANKS | 2 busy banks |
| 3 | 3BUSYBANKS | 3 busy banks |

**Bits 17:16 – CURBK_CTLDIR[1:0]** Current Bank/Control Direction (cleared upon USB reset)
**Control Direction (for Control endpoint only):**
0: A Control Write is requested by the Host.

1: A Control Read is requested by the Host.
**Note:**

1. Corresponds to the the 7th bit of the bmRequestType (Byte 0 of the Setup Data).
2. Updated after receiving new setup data.

**Current Bank (not relevant for Control endpoint):**

- Set by hardware to indicate the number of the current bank.
- Reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).
- The current bank is updated each time the user:
  - Sets the TX Packet Ready bit to prepare the next IN transfer and to switch to the next bank.
  - Clears the received OUT data bit to access the next bank.

| Value | Name | Description |
|---|---|---|
| 0 | BANK0 | Bank 0 (or single bank) |
| 1 | BANK1 | Bank 1 |
| 2 | BANK2 | Bank 2 |

**Bit 15 – NAK_OUT** NAK OUT (cleared upon USB reset)
This bit is set by hardware when a NAK handshake has been sent in response to an OUT or PING request from the Host.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by EPT_CTL_DISx (disable endpoint).

**Bit 14 – NAK_IN** NAK IN (cleared upon USB reset)
This bit is set by hardware when a NAK handshake has been sent in response to an IN request from the Host.

This bit is cleared by software.

**Bit 13 – STALL_SNT** Stall Sent (cleared upon USB reset)
For Control, Bulk and Interrupt endpoints.

This bit is set by hardware after a STALL handshake has been sent as requested by the UDPHS_EPTSTAx register FRCESTALL bit.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

**Bit 12 – RX_SETUP** Received SETUP (cleared upon USB reset)
For Control endpoint only.

This bit is set by hardware when a valid SETUP packet has been received from the host.

It is cleared by the device firmware after reading the SETUP data from the endpoint FIFO.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

**Bit 11 – TXRDY** TX Packet Ready (cleared upon USB reset)
This bit is cleared by hardware after the host has acknowledged the packet.

For Multi-bank endpoints, this bit may remain clear even after software is set if another bank is available to transmit.

Hardware clear of this bit may generate an interrupt if enabled by the UDPHS_EPTCTLx register TXRDY bit.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

**Bit 10 – TX_COMPLT** Transmitted IN Data Complete (cleared upon USB reset)
This bit is set by hardware after an IN packet has been accepted (ACK'ed) by the host.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

**Bit 9 – RXRDY_TXKL** Received OUT Data/KILL Bank (cleared upon USB reset)
**Received OUT Data (for OUT endpoint or Control endpoint):**

- This bit is set by hardware after a new packet has been stored in the endpoint FIFO.
- This bit is cleared by the device firmware after reading the OUT data from the endpoint.
- For multi-bank endpoints, this bit may remain active even when cleared by the device firmware, this if an other packet has been received meanwhile.
- Hardware assertion of this bit may generate an interrupt if enabled by the UDPHS_EPTCTLx register RXRDY_TXKL bit.
- This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

**KILL Bank (for IN endpoint):**

- The bank is really cleared or the bank is sent, BUSY_BANK_STA is decremented.
- The bank is not cleared but sent on the IN transfer, TX_COMPLT
- The bank is not cleared because it was empty. The user should wait that this bit is cleared before trying to clear another packet.
  **Note:** "Kill a packet" may be refused if at the same time, an IN token is coming and the current packet is sent on the UDPHS line. In this case, the TX_COMPLT bit is set. Take notice however, that if at least two banks are ready to be sent, there is no problem to kill a packet even if an IN token is coming. In fact, in that case, the current bank is sent (IN transfer) and the last bank is killed.

**Bit 8 – ERR_OVFLW** Overflow Error (cleared upon USB reset)
This bit is set by hardware when a new too-long packet is received.

Example: If the user programs an endpoint 64 bytes wide and the host sends 128 bytes in an OUT transfer, then the Overflow Error bit is set.

This bit is updated at the same time as the BYTE_COUNT field.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

**Bits 7:6 – TOGGLESQ_STA[1:0]** Toggle Sequencing (cleared upon USB reset)
In OUT transfer, the Toggle information is meaningful only when the current bank is busy (Received OUT Data = 1).

This field is updated for OUT transfer:

- A new data has been written into the current bank.
- The user has just cleared the Received OUT Data bit to switch to the next bank.

This field is reset to DATA1 by the UDPHS_EPTCLRSTAx register TOGGLESQ bit, and by UDPHS_EPTCTLDISx (disable endpoint).

Toggle Sequencing:

- IN endpoint: Indicates the PID Data Toggle that will be used for the next packet sent. This is not relative to the current bank.
- CONTROL and OUT endpoints: Set by hardware to indicate the PID data of the current bank.

| Value | Name | Description |
|---|---|---|
| 0 | DATA0 | DATA0 |
| 1 | DATA1 | DATA1 |
| 2 | DATA2 | Reserved for High Bandwidth Isochronous Endpoint |
| 3 | MDATA | Reserved for High Bandwidth Isochronous Endpoint |

**Bit 5 – FRCESTALL** Stall Handshake Request (cleared upon USB reset)
This bit is reset by hardware upon received SETUP.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If set a STALL answer will be done to the host for the next handshake. |

### 40.7.20 UDPHS Endpoint Status Register (Isochronous Endpoint)

**Name:** UDPHS_EPTSTAx
**Offset:** 0x011C + x*0x20 [x=0..15]
**Reset:** 0x00000040
**Property:** Read-only

This register view is relevant only if EPT_TYPE = 0x1 in “UDPHS Endpoint Configuration Register”.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SHRT_PCKT | BYTE_COUNT[10:4] | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BYTE_COUNT[3:0] | | | | BUSY_BANK_STA[1:0] | | CURBK[1:0] | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | ERR_FLUSH | ERR_CRC_NT | ERR_FL_ISO | TXRDY_TRER | TX_COMPLT | RXRDY_TXKL | ERR_OVFLW |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TOGGLESQ_STA[1:0] | | | | | | | |
| Access | R | R | | | | | | |
| Reset | 0 | 1 | | | | | | |

**Bit 31 – SHRT_PCKT** Short Packet (cleared upon USB reset)
An OUT Short Packet is detected when the receive byte count is less than the configured UDPHS_EPTCFGx register EPT_Size.

This bit is updated at the same time as the BYTE_COUNT field.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

**Bits 30:20 – BYTE_COUNT[10:0]** UDPHS Byte Count (cleared upon USB reset)
Byte count of a received data packet.

This field is incremented after each write into the endpoint (to prepare an IN transfer).

This field is decremented after each reading into the endpoint (OUT transfer).

This field is also updated at RXRDY_TXKL flag clear with the next bank.

This field is also updated at TXRDY_TRER flag set with the next bank.

This field is reset by EPT_x of UDPHS_EPTRST register.

**Bits 19:18 – BUSY_BANK_STA[1:0]** Busy Bank Number (cleared upon USB reset)
These bits are set by hardware to indicate the number of busy banks.

IN endpoint: It indicates the number of busy banks filled by the user, ready for IN transfer.

OUT endpoint: It indicates the number of busy banks filled by OUT transaction from the Host.

| Value | Name | Description |
|---|---|---|
| 0 | 0BUSYBANK | All banks are free |
| 1 | 1BUSYBANK | 1 busy bank |
| 2 | 2BUSYBANKS | 2 busy banks |
| 3 | 3BUSYBANKS | 3 busy banks |

**Bits 17:16 – CURBK[1:0]** Current Bank (cleared upon USB reset)
These bits are set by hardware to indicate the number of the current bank.

The current bank is updated each time the user:

- Sets the TX Packet Ready bit to prepare the next IN transfer and to switch to the next bank.
- Clears the received OUT data bit to access the next bank.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

| Value | Name | Description |
|---|---|---|
| 0 | BANK0 | Bank 0 (or single bank) |
| 1 | BANK1 | Bank 1 |
| 2 | BANK2 | Bank 2 |

**Bit 14 – ERR_FLUSH** Bank Flush Error (cleared upon USB reset)
For High Bandwidth Isochronous IN endpoints.

This bit is set when flushing unsent banks at the end of a microframe.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by EPT_CTL_DISx (disable endpoint).

**Bit 13 – ERR_CRC_NTR** CRC ISO Error/Number of Transaction Error (cleared upon USB reset)
**CRC ISO Error (for Isochronous OUT endpoints) (Read-only):**
This bit is set by hardware if the last received data is corrupted (CRC error on data).

This bit is updated by hardware when new data is received (Received OUT Data bit).

**Number of Transaction Error (for High Bandwidth Isochronous IN endpoints):**
This bit is set at the end of a microframe in which at least one data bank has been transmitted, if less than the number of transactions per micro-frame banks (UDPHS_EPTCFGx register NB_TRANS) have been validated for transmission inside this microframe.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

**Bit 12 – ERR_FL_ISO** Error Flow (cleared upon USB reset)
This bit is set by hardware when a transaction error occurs.

- Isochronous IN transaction is missed, the micro has no time to fill the endpoint (underflow).
- Isochronous OUT data is dropped because the bank is busy (overflow).

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

**Bit 11 – TXRDY_TRER** TX Packet Ready/Transaction Error (cleared upon USB reset)
**TX Packet Ready**
This bit is cleared by hardware, as soon as the packet has been sent.

For Multi-bank endpoints, this bit may remain clear even after software is set if another bank is available to transmit.

Hardware clear of this bit may generate an interrupt if enabled by the UDPHS_EPTCTLx register TXRDY_TRER bit.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

**Transaction Error (for high bandwidth isochronous OUT endpoints) (Read-Only):**
This bit is set by hardware when a transaction error occurs inside one microframe.

If one toggle sequencing problem occurs among the n-transactions (n = 1, 2 or 3) inside a microframe, then this bit is still set as long as the current bank contains one “bad” n-transaction (see CURBK field description). As soon as the current bank is relative to a new “good” n-transactions, then this bit is reset.

**Note:**

1. A transaction error occurs when the toggle sequencing does not comply with the Universal Serial Bus Specification, Rev 2.0 (5.9.2 High Bandwidth Isochronous endpoints) (Bad PID, missing data, etc.)
2. When a transaction error occurs, the user may empty all the “bad” transactions by clearing the Received OUT Data flag (RXRDY_TXKL).

If this bit is reset, then the user should consider that a new n-transaction is coming.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

**Bit 10 – TX_COMPLT** Transmitted IN Data Complete (cleared upon USB reset)
This bit is set by hardware after an IN packet has been sent.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

**Bit 9 – RXRDY_TXKL** Received OUT Data/KILL Bank (cleared upon USB reset)
**Received OUT Data (for OUT endpoint or Control endpoint):**

- This bit is set by hardware after a new packet has been stored in the endpoint FIFO.
- This bit is cleared by the device firmware after reading the OUT data from the endpoint.
- For multi-bank endpoints, this bit may remain active even when cleared by the device firmware, this if an other packet has been received meanwhile.
- Hardware assertion of this bit may generate an interrupt if enabled by the UDPHS_EPTCTLx register RXRDY_TXKL bit.
- This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

**KILL Bank (for IN endpoint):**

- The bank is really cleared or the bank is sent, BUSY_BANK_STA is decremented.

- The bank is not cleared but sent on the IN transfer, TX_COMPLT
- The bank is not cleared because it was empty. The user should wait that this bit is cleared before trying to clear another packet.
  **Note:** "Kill a packet" may be refused if at the same time, an IN token is coming and the current packet is sent on the UDPHS line. In this case, the TX_COMPLT bit is set. Take notice however, that if at least two banks are ready to be sent, there is no problem to kill a packet even if an IN token is coming. In fact, in that case, the current bank is sent (IN transfer) and the last bank is killed.

**Bit 8 – ERR_OVFLW** Overflow Error (cleared upon USB reset)
This bit is set by hardware when a new too-long packet is received.

Example: If the user programs an endpoint 64 bytes wide and the host sends 128 bytes in an OUT transfer, then the Overflow Error bit is set.

This bit is updated at the same time as the BYTE_COUNT field.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

**Bits 7:6 – TOGGLESQ_STA[1:0]** Toggle Sequencing (cleared upon USB reset)
In OUT transfer, the Toggle information is meaningful only when the current bank is busy (Received OUT Data = 1).

This field is updated for OUT transfer:

- A new data has been written into the current bank.
- The user has just cleared the Received OUT Data bit to switch to the next bank.

For High Bandwidth Isochronous Out endpoint, it is recommended to check the UDPHS_EPTSTAx/ TXRDY_TRER bit to know if the toggle sequencing is correct or not.

This field is reset to DATA1 by the UDPHS_EPTCLRSTAx register TOGGLESQ bit, and by UDPHS_EPTCTLDISx (disable endpoint).

Toggle Sequencing:

- IN endpoint: Indicates the PID Data Toggle that will be used for the next packet sent. This is not relative to the current bank.
- OUT endpoint: Set by hardware to indicate the PID data of the current bank:

| Value | Name | Description |
|---|---|---|
| 0 | DATA0 | DATA0 |
| 1 | DATA1 | DATA1 |
| 2 | DATA2 | Data2 (only for High Bandwidth Isochronous Endpoint) |
| 3 | MDATA | MData (only for High Bandwidth Isochronous Endpoint) |

### 40.7.21 UDPHS DMA Channel Transfer Descriptor

The DMA channel transfer descriptor is loaded from the memory. Be careful with the alignment of this buffer. The structure of the DMA channel transfer descriptor is defined by three parameters as described below:

- Offset 0:
  - The address must be aligned: 0xXXXX0
  - Next Descriptor Address Register: UDPHS_DMANXTDSCx

- Offset 4:
  - The address must be aligned: 0xXXXX4
  - DMA Channelx Address Register: UDPHS_DMAADDRESSx
- Offset 8:
  - The address must be aligned: 0xXXXX8
  - DMA Channelx Control Register: UDPHS_DMACONTROLx

To use the DMA channel transfer descriptor, fill the structures with the correct value (as described in the following pages). Then write directly in UDPHS_DMANXTDSCx the address of the descriptor to be used first. Then write '1' in the LDNXT_DSC bit of UDPHS_DMACONTROLx (load next channel transfer descriptor). The descriptor is automatically loaded upon Endpointx request for packet transfer.

### 40.7.22 UDPHS DMA Next Descriptor Address Register

**Name:** UDPHS_DMANXTDSCx
**Offset:** 0x0300 + x*0x10 [x=0..6]
**Reset:** 0x00000000
**Property:** Read/Write

Channel 0 is not used.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NXT_DSC_ADD[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NXT_DSC_ADD[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NXT_DSC_ADD[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NXT_DSC_ADD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NXT_DSC_ADD[31:0]** Next Descriptor Address
This field points to the next channel descriptor to be processed. This channel descriptor must be aligned, so bits 0 to 3 of the address must be equal to zero.

### 40.7.23 UDPHS DMA Channel Address Register

**Name:** UDPHS_DMAADDRESSx
**Offset:** 0x0304 + x*0x10 [x=0..6]
**Reset:** 0x00000000
**Property:** Read/Write

Channel 0 is not used.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_ADD[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_ADD[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_ADD[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_ADD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – BUFF_ADD[31:0]** Buffer Address
This field determines the AHB bus starting address of a DMA channel transfer.

Channel start and end addresses may be aligned on any byte boundary.

The firmware may write this field only when the UDPHS_DMASTATUS register CHANN_ENB bit is clear.

This field is updated at the end of the address phase of the current access to the AHB bus. It is incrementing of the access byte width. The access width is 4 bytes (or less) at packet start or end, if the start or end address is not aligned on a word boundary.

The packet start address is either the channel start address or the next channel address to be accessed in the channel buffer.

The packet end address is either the channel end address or the latest channel address accessed in the channel buffer.

The channel start address is written by software or loaded from the descriptor, whereas the channel end address is either determined by the end of buffer or the UDPHS device, USB end of transfer if the UDPHS_DMACONTROLx register END_TR_EN bit is set.

### 40.7.24 UDPHS DMA Channel Control Register

**Name:** UDPHS_DMACONTROLx
**Offset:** 0x0308 + x*0x10 [x=0..6]
**Reset:** 0x00000000
**Property:** Read/Write

Channel 0 is not used.

Bits [31:2] are only writable when issuing a channel Control Command other than “Stop Now”.

For reliability it is highly recommended to wait for both UDPHS_DMASTATUS.CHAN_ACT and CHAN_ENB flags at 0, thus ensuring the channel has been stopped before issuing a command other than “Stop Now”.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_LENGTH[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_LENGTH[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – BUFF_LENGTH[15:0]** Buffer Byte Length (Write-only)
This field determines the number of bytes to be transferred until end of buffer. The maximum channel transfer size (64 KBytes) is reached when this field is 0 (default value). If the transfer size is unknown, this field should be set to 0, but the transfer end may occur earlier under UDPHS device control.

When this field is written, the UDPHS_DMASTATUS.BUFF_COUNT field is updated with the write value.

**Bit 7 – BURST_LCK** Burst Lock Enable

| Value | Description |
|---|---|
| 0 | The DMA never locks bus access. |
| 1 | USB packets AHB data bursts are locked for maximum optimization of the bus bandwidth usage and maximization of fly-by AHB burst duration. |

**Bit 6 – DESC_LD_IT** Descriptor Loaded Interrupt Enable

| Value | Description |
|---|---|
| 0 | UDPHS_DMASTATUS.DESC_LDST rising will not trigger any interrupt. |
| 1 | An interrupt is generated when a descriptor has been loaded from the bus. |

**Bit 5 – END_BUFFIT** End of Buffer Interrupt Enable

| Value | Description |
|---|---|
| 0 | UDPHS_DMASTATUS.END_BF_ST rising will not trigger any interrupt. |
| 1 | An interrupt is generated when the UDPHS_DMASTATUS.BUFF_COUNT reaches zero. |

**Bit 4 – END_TR_IT** End of Transfer Interrupt Enable

| Value | Description |
|---|---|
| 0 | UDPHS device initiated buffer transfer completion will not trigger any interrupt at UDPHS_STATUS.END_TR_ST rising. |
| 1 | An interrupt is sent after the buffer transfer is complete, if the UDPHS device has ended the buffer transfer.<br>Use when the receive size is unknown. |

**Bit 3 – END_B_EN** End of Buffer Enable (Control)

| Value | Description |
|---|---|
| 0 | DMA Buffer End has no impact on USB packet transfer. |
| 1 | Endpoint can validate the packet (according to the values programmed in the UDPHS_EPTCTL.AUTO_VALID and SHRT_PCKT fields) at DMA Buffer End, i.e., when the UDPHS_DMASTATUS.BUFF_COUNT reaches 0.<br>This is mainly for short packet IN validation initiated by the DMA reaching end of buffer, but could be used for OUT packet truncation (discarding of unwanted packet data) at the end of DMA buffer. |

**Bit 2 – END_TR_EN** End of Transfer Enable (Control)
Used for OUT transfers only.

| Value | Description |
|---|---|
| 0 | USB end of transfer is ignored. |
| 1 | UDPHS device can put an end to the current buffer transfer.<br>When set, a BULK or INTERRUPT short packet or the last packet of an ISOCHRONOUS (micro) frame (DATAX) will close the current buffer and the UDPHS_DMASTATUS.END_TR_ST flag will be raised.<br>This is intended for UDPHS non-prenegotiated end of transfer (BULK or INTERRUPT) or ISOCHRONOUS microframe data buffer closure. |

**Bit 1 – LDNXT_DSC** Load Next Channel Transfer Descriptor Enable (Command)
0: No channel register is loaded after the end of the channel transfer.

1: The channel controller loads the next descriptor after the end of the current transfer, i.e., when the UDPHS_DMASTATUS/CHANN_ENB bit is reset.

If CHANN_ENB is cleared, the next descriptor is immediately loaded upon transfer request.

DMA Channel Control Command Summary

| LDNXT_DSC | CHANN_ENB | Description |
|---|---|---|
| 0 | 0 | Stop now |
| 0 | 1 | Run and stop at end of buffer |
| 1 | 0 | Load next descriptor now |
| 1 | 1 | Run and link at end of buffer |

**Bit 0 – CHANN_ENB** (Channel Enable Command)

| Value | Description |
|---|---|
| 0 | DMA channel is disabled at and no transfer will occur upon request. This bit is also cleared by hardware when the channel source bus is disabled at end of buffer.<br>If the UDPHS_DMACONTROL register LDNXT_DSC bit has been cleared by descriptor loading, the firmware will have to set the corresponding CHANN_ENB bit to start the described transfer, if needed.<br>If the UDPHS_DMACONTROL register LDNXT_DSC bit is cleared, the channel is frozen and the channel registers may then be read and/or written reliably as soon as both UDPHS_DMASTATUS register CHANN_ENB and CHANN_ACT flags read as 0.<br>If a channel request is currently serviced when this bit is cleared, the DMA FIFO buffer is drained until it is empty, then the UDPHS_DMASTATUS register CHANN_ENB bit is cleared.<br>If the LDNXT_DSC bit is set at or after this bit clearing, then the currently loaded descriptor is skipped (no data transfer occurs) and the next descriptor is immediately loaded. |
| 1 | UDPHS_DMASTATUS register CHANN_ENB bit will be set, thus enabling DMA channel data transfer. Then any pending request will start the transfer. This may be used to start or resume any requested transfer. |

### 40.7.25 UDPHS DMA Channel Status Register

**Name:** UDPHS_DMASTATUSx
**Offset:** 0x030C + x*0x10 [x=0..6]
**Reset:** 0x00000000
**Property:** Read/Write

Channel 0 is not used.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_COUNT[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_COUNT[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | | | 0 | 0 |

**Bits 31:16 – BUFF_COUNT[15:0]** Buffer Byte Count
This field determines the current number of bytes still to be transferred for this buffer.

This field is decremented from the AHB source bus access byte width at the end of this bus address phase.

The access byte width is 4 by default, or less, at DMA start or end, if the start or end address is not aligned on a word boundary.

At the end of buffer, the DMA accesses the UDPHS device only for the number of bytes needed to complete it.

This field value is reliable (stable) only if the channel has been stopped or frozen (UDPHS_EPTCTLx register NT_DIS_DMA bit is used to disable the channel request) and the channel is no longer active CHANN_ACT flag is 0.

**Note:** For OUT endpoints, if the receive buffer byte length (BUFF_LENGTH) has been defaulted to zero because the USB transfer length is unknown, the actual buffer byte length received is 0x10000-BUFF_COUNT.

**Bit 6 – DESC_LDST** Descriptor Loaded Status
Valid until the CHANN_ENB flag is cleared at the end of the next buffer transfer.

| Value | Description |
|---|---|
| 0 | Cleared automatically when read by software. |
| 1 | Set by hardware when a descriptor has been loaded from the system bus. |

**Bit 5 – END_BF_ST** End of Channel Buffer Status
Valid until the CHANN_ENB flag is cleared at the end of the next buffer transfer.

| Value | Description |
|---|---|
| 0 | Cleared automatically when read by software. |
| 1 | Set by hardware when the BUFF_COUNT countdown reaches zero. |

**Bit 4 – END_TR_ST** End of Channel Transfer Status
Valid until the CHANN_ENB flag is cleared at the end of the next buffer transfer.

| Value | Description |
|---|---|
| 0 | Cleared automatically when read by software. |
| 1 | Set by hardware when the last packet transfer is complete, if the UDPHS device has ended the transfer. |

**Bit 1 – CHANN_ACT** Channel Active Status
When a packet transfer is ended this bit is automatically reset.

When a packet transfer cannot be completed due to an END_BF_ST, this flag stays set during the next channel descriptor load (if any) and potentially until UDPHS packet transfer completion, if allowed by the new descriptor.

| Value | Description |
|---|---|
| 0 | The DMA channel is no longer trying to source the packet data. |
| 1 | The DMA channel is currently trying to source packet data, i.e., selected as the highest-priority requesting channel. |

**Bit 0 – CHANN_ENB** Channel Enable Status
When any transfer is ended either due to an elapsed byte count or a UDPHS device initiated transfer end, this bit is automatically reset.

This bit is normally set or cleared by writing into the UDPHS_DMACONTROLx register CHANN_ENB bit either by software or descriptor loading.

If a channel request is currently serviced when the UDPHS_DMACONTROLx register CHANN_ENB bit is cleared, the DMA FIFO buffer is drained until it is empty, then this status bit is cleared.

| Value | Description |
|---|---|
| 0 | The DMA channel no longer transfers data, and may load the next descriptor if the UDPHS_DMACONTROLx register LDNXT_DSC bit is set. |
| 1 | The DMA channel is currently enabled and transfers data upon request. |

# 41. USB Host High Speed Port (UHPHS)

## 41.1 Description

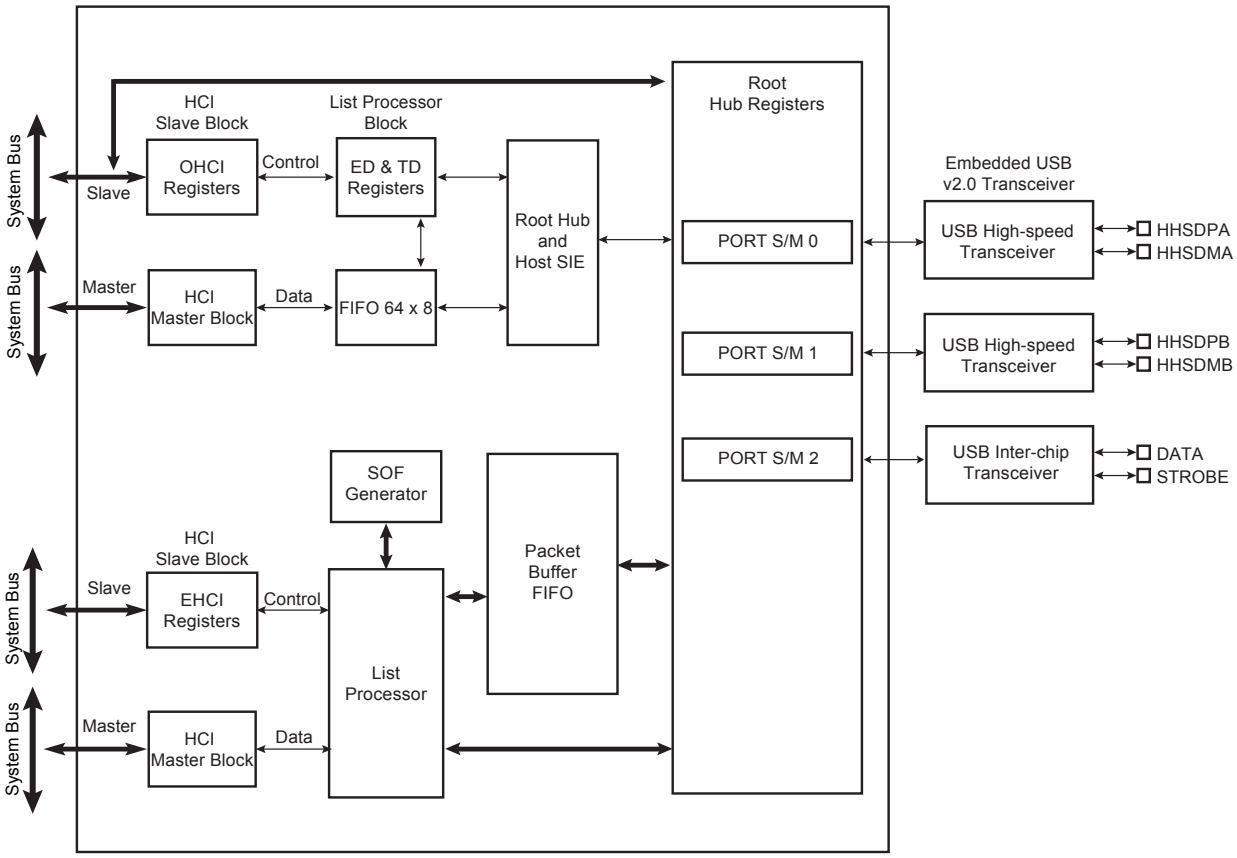
The USB Host High Speed Port (UHPHS) interfaces the USB with the host application. It handles Open HCI protocol (Open Host Controller Interface) as well as Enhanced HCI protocol (Enhanced Host Controller Interface).

## 41.2 Embedded Characteristics

- Compliant with Enhanced HCI Rev 1.0 Specification
  - Compliant with USB V2.0 High-speed
  - Supports High-speed 480 Mbps
- Compliant with Open HCI Rev 1.0 Specification
  - Compliant with USB V2.0 Full-speed and Low-speed Specification
  - Supports both Low-speed 1.5 Mbps and Full-speed 12 Mbps USB devices
- Root Hub Integrated with 3 Downstream USB HS Ports
- Embedded USB Transceivers
- Supports Power Management
- 3 Hosts (A, B and C) High Speed (EHCI), Port A shared with UDPHS
  - 1 Host (C) High Speed Inter-Chip Only (HSIC)

## 41.3 Block Diagram

**Figure 41-1. Block Diagram**




Access to the USB host operational registers is achieved through the system bus slave interface. The Open HCI host controller and Enhanced HCI host controller initialize master DMA transfers through the system bus master interface as follows:

- Fetches endpoint descriptors and transfer descriptors
- Access to endpoint data from system memory
- Access to the HC communication area
- Write status and retire transfer descriptor

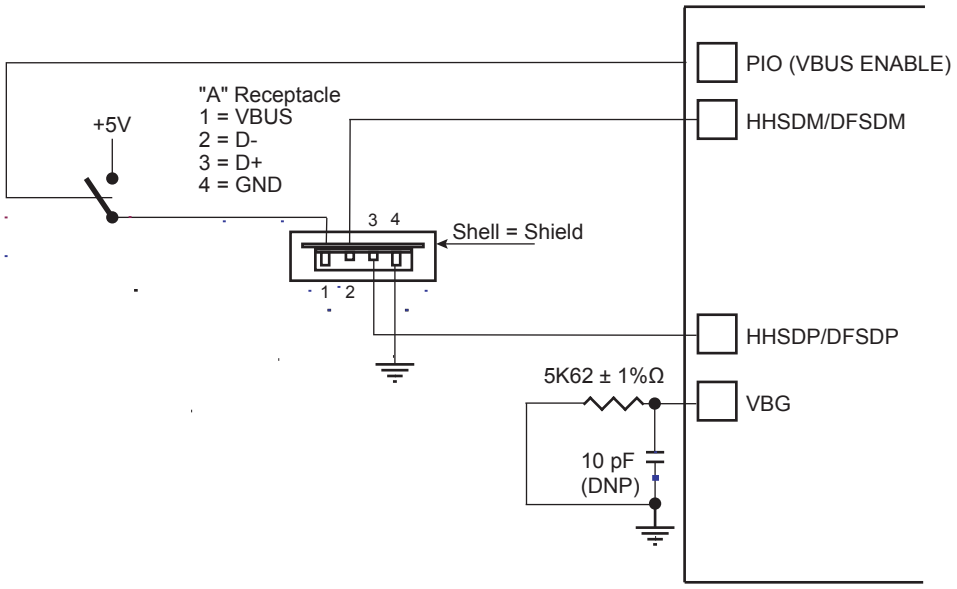
Memory access errors (abort, misalignment) lead to an “Unrecoverable Error” indicated by the corresponding flag in the host controller operational registers.

The USB root hub is integrated in the USB host. Several USB downstream ports are available. The number of downstream ports can be determined by the software driver reading the root hub’s operational registers. Device connection is automatically detected by the USB host port logic.

USB physical transceivers are integrated in the product and driven by the root hub’s ports.

## 41.4 Typical Connection

**Figure 41-2. Board Schematic to Interface UHP High-speed Host Controller**




## 41.5 Product Dependencies

### 41.5.1 I/O Lines

HFSDPs, HFSDMs, HHSDPs and HHSDMs are not controlled by any PIO controllers. The embedded USB High Speed physical transceivers are controlled by the USB host controller.

One transceiver is shared with the USB High Speed Device (port A). The selection between Host Port A and USB Device is controlled by the UDPHS enable bit (EN_UDPHS) located in the UDPHS_CTRL register.

In the case the port A is driven by the USB High Speed Device, the output signals are DFSDP, DFSDM, DHSDP and DHSDM. The transceiver is automatically selected for Device operation once the USB High Speed Device is enabled.

In the case the port A is driven by the USB High Speed Host, the output signals are HFSDPA, HFSDMA, HHSDPA and HHSDMA.

### 41.5.2 Power Management

The system embeds 3 transceivers.

The USB Host High Speed requires a 480 MHz clock for the embedded High-speed transceivers. This clock (UPLLCK) is provided by the UTMI PLL.

In case power consumption is saved by stopping the UTMI PLL, high-speed operations are not possible. Nevertheless, OHCI Full-speed operations remain possible by selecting PLLACK as the input clock of OHCI.

The High-speed transceiver returns a 30 MHz clock to the USB Host controller.

The USB Host controller requires 48 MHz and 12 MHz clocks for OHCI full-speed operations. These clocks must be generated by a PLL with a correct accuracy of +/-0.25% using the USBDIV field.

Thus the USB Host peripheral receives three clocks from the Power Management Controller (PMC): the Peripheral Clock (MCK domain), the UHP48M and the UHP12M (built-in UHP48M divided by four) used by the OHCI to interface with the bus USB signals (recovered 12 MHz domain) in Full-speed operations.

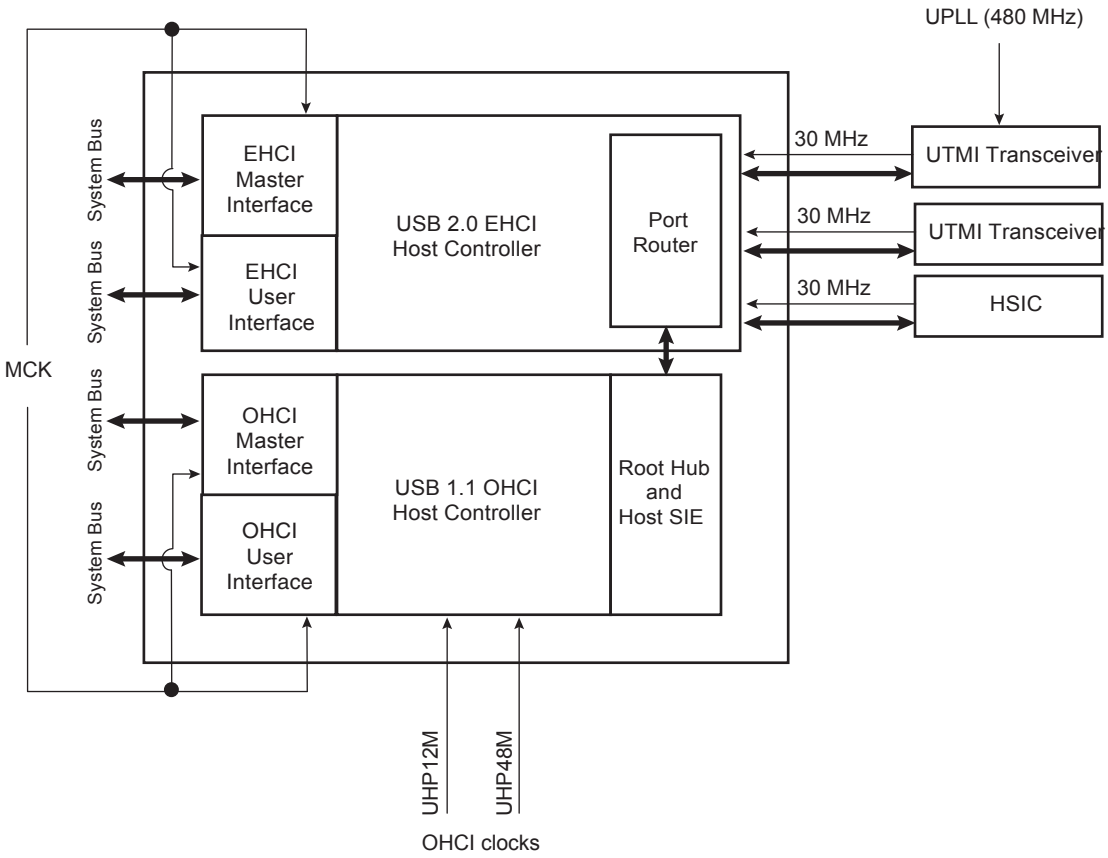
For High-speed operations, the user has to perform the following:

- Enable UHP peripheral clock in PMC_PCER.
- Write PLLCOUNT field in CKGR_UCKR.
- Enable UPLL with UPLLEN bit in CKGR_UCKR.
- Wait until UTMI_PLL is locked (LOCKU bit in PMC_SR).
- Select UPLLCK as Input clock of OHCI part (USBS bit in PMC_USB register).
- Program OHCI clocks (UHP48M and UHP12M) with USBDIV field in PMC_USB register. USBDIV must be 9 (division by 10) if UPLLCK is selected.
- Enable OHCI clocks with UHP bit in PMC_SCER.

For OHCI Full-speed operations only, the user has to perform the following:

- Enable UHP peripheral clock in PMC_PCER.
- Select PLLACK as Input clock of OHCI part (USBS bit in PMC_USB register).
- Program OHCI clocks (UHP48M and UHP12M) with USBDIV field in PMC_USB register. USBDIV value is to be calculated according to the PLLACK value and USB Full-speed accuracy.
- Enable the OHCI clocks with UHP bit in PMC_SCER.

**Figure 41-3. UHP Clock Trees**



### 41.5.3 Interrupt Sources

The USB host interface has an interrupt line connected to the interrupt controller.

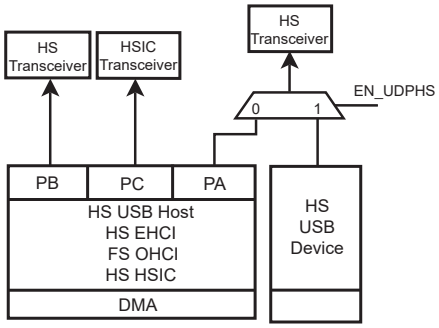
Handling USB host interrupts requires programming the interrupt controller before configuring the UHPHS.

## 41.6 Functional Description

### 41.6.1 UTMI Transceivers Sharing

The High Speed USB Host Port A is shared with the High Speed USB Device port and connected to the second UTMI transceiver. The selection between Host Port A and USB device is controlled by the UDPHS enable bit (EN_UDPHS) located in the UDPHS_CTRL register.

**Figure 41-4. USB Selection**




### 41.6.2 EHCI

The USB Host Port controller is fully compliant with the Enhanced HCI specification. The USB Host Port User Interface (registers description) can be found in the Enhanced HCI Rev 1.0 Specification available on www.usb.org

### 41.6.3 OHCI

The USB Host Port integrates a root hub and transceivers on downstream ports. It provides several Full-speed half-duplex serial communication ports at a baud rate of 12 Mbps. Up to 127 USB devices (printer, camera, mouse, keyboard, disk, etc.) and the USB hub can be connected to the USB host in the USB “tiered star” topology.

The USB Host Port controller is fully compliant with the Open HCI specification. The USB Host Port User Interface (registers description) can be found in the Open HCI Rev 1.0 Specification available on www.usb.org.

All standard class devices are automatically detected and available to the user’s application. As an example, integrating an HID (Human Interface Device) class driver provides a plug & play feature for all USB keyboards and mouses.

### 41.6.4 HSIC

The High-Speed Inter-Chip (HSIC) is a standard for USB chip-to-chip interconnect with a 2-signal (strobe, data) source synchronous serial interface using 240 MHz DDR signaling to provide only high-speed 480 Mbps data rate.

External cables, connectors and hot plug & play are not supported.

The HSIC interface operates at high speed, 480 Mbps, and is fully compatible with existing USB software stacks. It meets all data transfer needs through a single unified USB software stack.

## 41.7 Register Summary

The Enhanced USB Host Controller contains two sets of software-accessible hardware registers: memory-mapped Host Controller Registers and optional PCI configuration registers. Note that the PCI configuration registers are only needed for PCI devices that implement the Host Controller.

- Memory-mapped USB Host Controller Registers—This block of registers is memory-mapped into non-cacheable memory. This memory space must begin on a DWord (32-bit) boundary. This register space is divided into two sections: a set of read-only capability registers and a set of read/write operational registers. The table below describes each register space.
  **Note:** Host controllers are not required to support exclusive-access mechanisms (such as PCI LOCK) for accesses to the memory-mapped register space. Therefore, if software attempts exclusive-access mechanisms to the host controller memory-mapped register space, the results are undefined.
- PCI Configuration Registers (for PCI devices)—In addition to the normal PCI header, power management, and device-specific registers, two registers are needed in the PCI configuration space to support USB. The normal PCI header and device-specific registers are beyond the scope of this document (the UHPHS_CLASSC register is shown in this document). Note that HCD does not interact with the PCI configuration space. This space is used only by the PCI enumerator to identify the USB Host Controller, and assign the appropriate system resources.
  The table below summarizes the enhanced interface register sets.

| Offset | Register Set | Explanation |
|---|---|---|
| 0 to N-1 | Capability Registers | The capability registers specify the limits, restrictions, and capabilities of a host controller implementation.<br>These values are used as parameters to the host controller driver. |
| N to N+M-1 | Operational Registers | The operational registers are used by system software to control and monitor the operational state of the host controller. |

**Note:** Software must not modify reserved bits in Read/Write registers.

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | UHPHS_HCCAPBASE | 7:0 | CAPLENGTH[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | HCIVERSION[7:0] | | | | | | | |
| | | 31:24 | HCIVERSION[15:8] | | | | | | | |
| 0x04 | UHPHS_HCSPARAMS | 7:0 | | | | PPC | N_PORTS[3:0] | | | |
| | | 15:8 | N_CC[3:0] | | | | N_PCC[3:0] | | | |
| | | 23:16 | N_DP[3:0] | | | | | | | P_INDICATOR |
| | | 31:24 | | | | | | | | |
| 0x08 | UHPHS_HCCPARAMS | 7:0 | IST[3:0] | | | | | ASPC | PFLF | AC |
| | | 15:8 | EECP[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C ... 0x0F | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x10 | UHPHS_USBCMD | 7:0 | LHCR | IAAD | ASE | PSE | FLS[1:0] | | HCRESET | RS |
| | | 15:8 | | | | | ASPME | | ASPMC[1:0] | |
| | | 23:16 | ITC[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | UHPHS_USBSTS | 7:0 | | | IAA | HSE | FLR | PCD | USBERRINT | USBINT |
| | | 15:8 | ASS | PSS | RCM | HCHLT | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | UHPHS_USBINTR | 7:0 | | | IAAE | HSEE | FLRE | PCIE | USBEIE | USBIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | UHPHS_FRINDEX | 7:0 | FI[7:0] | | | | | | | |
| | | 15:8 | | | FI[13:8] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 ... 0x23 | Reserved | | | | | | | | | |
| 0x24 | UHPHS_PERIODICLISTBASE | 7:0 | | | | | | | | |
| | | 15:8 | BA[3:0] | | | | | | | |
| | | 23:16 | BA[11:4] | | | | | | | |
| | | 31:24 | BA[19:12] | | | | | | | |
| 0x28 | UHPHS_ASYNCLISTADDR | 7:0 | LPL[2:0] | | | | | | | |
| | | 15:8 | LPL[10:3] | | | | | | | |
| | | 23:16 | LPL[18:11] | | | | | | | |
| | | 31:24 | LPL[26:19] | | | | | | | |
| 0x2C ... 0x4F | Reserved | | | | | | | | | |
| 0x50 | UHPHS_CONFIGFLAG | 7:0 | | | | | | | | CF |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x54 | UHPHS_PORTSC0 | 7:0 | SUS | FPR | OCC | OCA | PEDC | PED | CSC | CCS |
| | | 15:8 | PIC[1:0] | | PO | PP | LS[1:0] | | | PR |
| | | 23:16 | | WKOC_E | WKDSCNNT_E | WKCNNT_E | PTC[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x58 | UHPHS_PORTSC1 | 7:0 | SUS | FPR | OCC | OCA | PEDC | PED | CSC | CCS |
| | | 15:8 | PIC[1:0] | | PO | PP | LS[1:0] | | | PR |
| | | 23:16 | | WKOC_E | WKDSCNNT_E | WKCNNT_E | PTC[3:0] | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x5C | UHPHS_PORTSC2 | 7:0 | SUS | FPR | OCC | OCA | PEDC | PED | CSC | CCS |
| | | 15:8 | PIC[1:0] | | PO | PP | LS[1:0] | | | PR |
| | | 23:16 | | WKOC_E | WKDSCNNT_E | WKCNNT_E | PTC[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x60 ... 0xA7 | Reserved | | | | | | | | | |
| 0xA8 | UHPHS_INSNREG06 | 7:0 | Nb_Burst[3:0] | | | | Nb_Success_Burst[3:0] | | | |
| | | 15:8 | | | | | HBURST[2:0] | | | Nb_Burst[4:4] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | AHB_ERR | | | | | | | |
| 0xAC | UHPHS_INSNREG07 | 7:0 | AHB_ADDR[7:0] | | | | | | | |
| | | 15:8 | AHB_ADDR[15:8] | | | | | | | |
| | | 23:16 | AHB_ADDR[23:16] | | | | | | | |
| | | 31:24 | AHB_ADDR[31:24] | | | | | | | |
| 0xB0 | UHPHS_INSNREG08 | 7:0 | | | | | | HSIC_EN | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 41.7.1 UHPHS Host Controller Capability Register

**Name:** UHPHS_HCCAPBASE
**Offset:** 0x00
**Reset:** 0x01000010
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | HCIVERSION[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | HCIVERSION[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CAPLENGTH[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

**Bits 31:16 – HCIVERSION[15:0]** Host Controller Interface Version Number
This is a two-byte field containing a BCD encoding of the EHCI revision number supported by this host controller. The most significant byte of this field represents the major revision and the least significant byte the minor revision.

**Bits 7:0 – CAPLENGTH[7:0]** Capability Registers Length
This field is used as an offset to add to the register base to find the beginning of the Operational Register Space.

### 41.7.2 UHPHS Host Controller Structural Parameters Register

**Name:** UHPHS_HCSPARAMS
**Offset:** 0x04
**Reset:** 0x00001116
**Property:** Read-only

These fields define structural parameters: number of downstream ports, etc.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | N_DP[3:0] | | | | | | | P_INDICATOR |
| Access | R | R | R | R | | | | R |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | N_CC[3:0] | | | | N_PCC[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | PPC | N_PORTS[3:0] | | | |
| Access | | | | R | R | R | R | R |
| Reset | | | | 1 | 0 | 1 | 1 | 0 |

**Bits 23:20 – N_DP[3:0]** Debug Port Number
Optional. This register identifies which of the host controller ports is the debug port. The value is the port number (1-based) of the debug port. A non-zero value in this field indicates the presence of a debug port. The value in this register must not be greater than N_PORTS (see above).

**Bit 16 – P_INDICATOR** Port Indicators
This bit indicates whether the ports support port indicator control. When this bit is a 1, the port status and control registers include a read/writeable field for controlling the state of the port indicator. See UHPHS Port Status and Control Register for a definition of the port indicator control field.

**Bits 15:12 – N_CC[3:0]** Number of Companion Controllers
This field indicates the number of companion controllers associated with this USB 2.0 host controller.

A zero in this field indicates there are no companion host controllers. Port-ownership hand-off is not supported. Only high-speed devices are supported on the host controller root ports.

A value larger than zero in this field indicates there are companion USB 1.1 host controller(s). Port-ownership hand-offs are supported. High, Full- and Low-speed devices are supported on the host controller root ports.

**Bits 11:8 – N_PCC[3:0]** Number of Ports per Companion Controller
This field indicates the number of ports supported per companion host controller. It is used to indicate the port routing configuration to system software.

For example, if N_PORTS has a value of 6 and N_CC has a value of 2, then N_PCC could have a value of 3. The convention is that the first N_PCC ports are assumed to be routed to companion controller 1, the next N_PCC ports to companion controller 2, etc. In the previous example, the N_PCC could have been 4, where the first four are routed to companion controller 1 and the last two are routed to companion controller 2.

The number in this field must be consistent with N_PORTS and N_CC.

**Bit 4 – PPC** Port Power Control
This field indicates whether the host controller implementation includes port power control. A one in this bit indicates the ports have port power switches. A zero in this bit indicates the ports do not have port power switches. The value of this field affects the functionality of the Port Power field in each port status and control register (see UHPHS Port Status and Control Register).

**Bits 3:0 – N_PORTS[3:0]** Number of Ports
This field specifies the number of physical downstream ports implemented on this host controller. The value of this field determines how many port registers are addressable in the Operational Register Space. Valid values are in the range of 1 to 15.

A zero in this field is undefined.

### 41.7.3 UHPHS Host Controller Capability Parameters Register

**Name:** UHPHS_HCCPARAMS
**Offset:** 0x08
**Reset:** 0x0000A010
**Property:** Read-only

These fields define capability parameters: Multiple Mode control (time-base bit functionality), addressing capability, etc.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EECP[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IST[3:0] | | | | | ASPC | PFLF | AC |
| Access | R | R | R | R | | R | R | R |
| Reset | 0 | 0 | 0 | 1 | | 0 | 0 | 0 |

**Bits 15:8 – EECP[7:0]** EHCI Extended Capabilities Pointer
Indicates the existence of a capabilities list. A value of 0 indicates no extended capabilities are implemented. A non-zero value in this register indicates the offset in the PCI configuration space of the first EHCI extended capability. The pointer value must be 64 or greater if implemented to maintain the consistency of the PCI header defined for this class of device.

**Bits 7:4 – IST[3:0]** Isochronous Scheduling Threshold
Indicates, relative to the current position of the executing host controller, where software can reliably update the isochronous schedule. When bit [7] is 0, the value of the least significant three bits indicates the number of microframes a host controller can hold a set of isochronous data structures (one or more) before flushing the state. When bit [7] is set to 1, the host software assumes the host controller may cache an isochronous data structure for an entire frame.

**Bit 2 – ASPC** Asynchronous Schedule Park Capability
The park capability can be disabled or enabled and set to a specific level by using the Asynchronous Schedule Park Mode Enable and Asynchronous Schedule Park Mode Count fields in the UHPHS_USBCMD register.

| Value | Description |
|---|---|
| 0 | Host controller does not supports the park feature for high-speed queue. |

| Value | Description |
|---|---|
| 1 | Host controller supports the park feature for high-speed queue heads in the Asynchronous Schedule. |

**Bit 1 – PFLF** Programmable Frame List Flag

| Value | Description |
|---|---|
| 0 | System software must use a frame list length of 1024 elements with this host controller. The UHPHS_USBCMD register Frame List Size field is a read-only register and must be set to 0. |
| 1 | System software can specify and use a smaller frame list and configure the host controller via the UHPHS_USBCMD register Frame List Size field. The frame list must always be aligned on a 4-Kbyte page boundary. This requirement ensures that the frame list is always physically contiguous. |

**Bit 0 – AC** 64-bit Addressing Capability
This field documents the addressing range capability of this implementation. The value of this field determines whether software should use 32-bit or 64-bit data structures.

This information is not tightly coupled with the UHPHS_USBBASE address register mapping control. The 64-bit Addressing Capability bit indicates whether the host controller can generate 64-bit addresses as a master. The UHPHS_USBBASE register indicates the host controller only needs to decode 32-bit addresses as a slave.

| Value | Description |
|---|---|
| 0 | Data structures using 32-bit address memory pointers |
| 1 | Data structures using 64-bit address memory pointers |

### 41.7.4 UHPHS USB Command Register

**Name:** UHPHS_USBCMD
**Offset:** 0x10
**Reset:** 0x00080B00
**Property:** Read/Write

The Command Register indicates the command to be executed by the serial bus host controller. Writing to the register causes a command to be executed.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ITC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | ASPME | | ASPMC[1:0] | |
| Access | | | | | R-R/W | | R-R/W | R-R/W |
| Reset | | | | | 1 | | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LHCR | IAAD | ASE | PSE | FLS[1:0] | | HCRESET | RS |
| Access | R/W | R/W | R/W | R/W | R-R/W | R-R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – ITC[7:0]** Interrupt Threshold Control
This field is used by system software to select the maximum rate at which the host controller will issue interrupts. The only valid values are defined below. If software writes an invalid value to this register, the results are undefined.

| Value | Maximum Interrupt Interval |
|---|---|
| 0 | Reserved |
| 1 | 1 microframe |
| 2 | 2 microframes |
| 4 | 4 microframes |
| 8 | 8 microframes (1 ms) |
| 16 | 16 microframes (2 ms) |
| 32 | 32 microframes (4 ms) |
| 64 | 64 microframes (8 ms) |

Any other value in this register yields undefined results.

Software modifications to this field while HCHLT=0 results in undefined behavior.

**Bit 11 – ASPME** Asynchronous Schedule Park Mode Enable (optional)
If the Asynchronous Park Capability bit in the UHPHS_HCCPARAMS register is set to 1, then this bit is set to 1 and is Read/Write. Otherwise the bit must be 0 and is read-only.

| Value | Description |
|---|---|
| 0 | Park mode is enabled. |
| 1 | Park mode is disabled. |

**Bits 9:8 – ASPMC[1:0]** Asynchronous Schedule Park Mode Count (optional)
If the Asynchronous Park Capability bit in the UHPHS_HCCPARAMS register is set to 1, then this field defaults to 3 and is read/write. Otherwise it defaults to 0 and is read-only. It contains a count of the number of successive transactions the host controller is allowed to execute from a high-speed queue head on the Asynchronous schedule before continuing traversal of the Asynchronous schedule. Valid values are 1 to 3. Software must not write a 0 to this bit when Park Mode Enable is set to 1 as this will result in undefined behavior.

**Bit 7 – LHCR** Light Host Controller Reset (optional)
This control bit is not required. If implemented, it allows the driver to reset the EHCI controller without affecting the state of the ports or the relationship to the companion host controllers. For example, the UHPHS_PORTSC registers should not be reset to their default values and the CF bit setting should not go to 0 (retaining port ownership relationships).

A host software read of this bit as 0 indicates the Light Host Controller Reset has completed and it is safe for host software to re-initialize the host controller. A host software read of this bit as 1 indicates the Light Host Controller Reset has not yet completed.

If not implemented, a read of this field will always return a 0.

**Bit 6 – IAAD** Interrupt on Async Advance Doorbell
This bit is used as a doorbell by software to tell the host controller to issue an interrupt the next time it advances asynchronous schedule. Software must write a 1 to this bit to ring the doorbell.

When the host controller has evicted all appropriate cached schedule state, it sets the Interrupt on Async Advance status bit in the UHPHS_USBSTS register. If the Interrupt on Async Advance Enable bit in the UHPHS_USBINTR register is set to 1, then the host controller will assert an interrupt at the next interrupt threshold.

The host controller sets this bit to 0 after it has set the Interrupt on Async Advance status bit in the UHPHS_USBSTS register to 1.

Software should not write a 1 to this bit when the asynchronous schedule is disabled. Doing so will yield undefined results.

**Bit 5 – ASE** Asynchronous Schedule Enable
This bit controls whether the host controller skips processing the Asynchronous Schedule.

| Value | Description |
|---|---|
| 0 | Do not process the Asynchronous Schedule. |
| 1 | Use the UHPHS_ASYNCLISTADDR register to access the Asynchronous Schedule. |

**Bit 4 – PSE** Periodic Schedule Enable
This bit controls whether the host controller skips processing the Periodic Schedule.

| Value | Description |
|---|---|
| 0 | Do not process the Periodic Schedule. |
| 1 | Use the UHPHS_PERIODICLISTBASE register to access the Periodic Schedule. |

**Bits 3:2 – FLS[1:0]** Frame List Size
This field is read-only with one exception: it is read/write if the Programmable Frame List flag, in the UHPHS_HCCPARAMS register, is set to 1. This field specifies the size of the frame list. The size of the frame list controls which bits in the Frame Index Register should be used for the Frame List Current index.

| Value | Description |
|---|---|
| 0 | 1024 elements (4096 bytes). |
| 1 | 512 elements (2048 bytes). |
| 2 | 256 elements (1024 bytes), for resource-constrained environments. |
| 3 | Reserved. |

**Bit 1 – HCRESET** Host Controller Reset
This control bit is used by software to reset the host controller. The effects of this on Root Hub registers are similar to a Chip Hardware Reset.

When software writes a 1 to this bit, the Host Controller resets its internal pipelines, timers, counters, state machines, etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. A USB reset is not driven on downstream ports.

PCI Configuration registers are not affected by this reset. All operational registers, including port registers and port state machines, are set to their initial values. Port ownership reverts to the companion host controller(s) with side effects. Software must reinitialize the host controller in order to return the host controller to an operational state.

This bit is set to 0 by the Host Controller when the reset process is complete. Software cannot terminate the reset process early by writing a 0 to this register.

Software must not set this bit to 1 when HCHLT in the UHPHS_USBSTS register is 0. Attempting to reset an actively running host controller results in undefined behavior.

**Bit 0 – RS** Run/Stop
The Host Controller must halt within 16 microframes after software clears the bit RS. The HCHLT bit in the status register indicates when the Host Controller has finished its pending pipelined transactions and has entered the stopped state. Software must not write 1 to this field unless the host controller is in the halted state (i.e., HCHLT in the UHPHS_USBSTS register is 1). Doing so yields undefined results.

| Value | Description |
|---|---|
| 0 | Host Controller completes the current and any actively pipelined transactions on the USB and then halts. |
| 1 | Host Controller proceeds with execution of the schedule. |

### 41.7.5 UHPHS USB Status Register

**Name:** UHPHS_USBSTS
**Offset:** 0x14
**Reset:** 0x00001000
**Property:** Read/Write

This register indicates pending interrupts and various states of the Host Controller. The status resulting from a transaction on the serial bus is not indicated in this register. Software sets a bit to 0 in this register by writing a 1 to it.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ASS | PSS | RCM | HCHLT | | | | |
| Access | R | R | R | R | | | | |
| Reset | 0 | 0 | 0 | 1 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | IAA | HSE | FLR | PCD | USBERRINT | USBINT |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 15 – ASS** Asynchronous Schedule Status
The bit reports the current real status of the Asynchronous Schedule.

The Host Controller is not required to immediately disable or enable the Asynchronous Schedule when software transitions the Asynchronous Schedule Enable bit in the UHPHS_USBCMD register. When this bit and the Asynchronous Schedule Enable bit are the same value, the Asynchronous Schedule is either enabled or disabled.

| Value | Description |
|---|---|
| 0 | Asynchronous Schedule is disabled. |
| 1 | Asynchronous Schedule is enabled. |

**Bit 14 – PSS** Periodic Schedule Status
The bit reports the current real status of the Periodic Schedule. If this bit is set to 0, then the status of the Periodic Schedule is disabled. If this bit is set to 1, then the status of the Periodic Schedule is enabled. The Host Controller is not required to immediately disable or enable the Periodic Schedule when software transitions the Periodic Schedule Enable bit in the UHPHS_USBCMD register. When this bit and the Periodic Schedule Enable bit are the same value, the Periodic Schedule is either enabled or disabled.

**Bit 13 – RCM** Reclamation
This is a read-only status bit used to detect any empty asynchronous schedule.

**Bit 12 – HCHLT** HCHalted
This bit is 0 whenever the Run/Stop bit is 1. The Host Controller sets this bit to 1 after it has stopped executing as a result of the Run/Stop bit being set to 0, either by software or by the Host Controller hardware (e.g. internal error).

**Bit 5 – IAA** Interrupt on Async Advance (Cleared on write)
System software can force the host controller to issue an interrupt the next time the host controller advances the asynchronous schedule by writing 1 to the Interrupt on the Async Advance Doorbell bit in the UHPHS_USBCMD register. This status bit indicates the assertion of that interrupt source.

**Bit 4 – HSE** Host System Error (Cleared on write)
The Host Controller sets this bit to 1 when a serious error occurs during a host system access involving the Host Controller module. In a PCI system, conditions that set this bit to 1 include PCI Parity error, PCI Master Abort, and PCI Target Abort. When this error occurs, the Host Controller clears the Run/Stop bit in the Command register to prevent further execution of the scheduled TDs.

**Bit 3 – FLR** Frame List Rollover (Cleared on write)
The Host Controller sets this bit to 1 when the Frame List Index (see UHPHS USB Frame Index Register) rolls over from its maximum value to 0. The exact value at which the rollover occurs depends on the frame list size. For example, if the frame list size (as programmed in the Frame List Size field of the UHPHS_USBCMD register) is 1024, the Frame Index Register rolls over every time FRINDEX[13] toggles. Similarly, if the size is 512, the Host Controller sets this bit to 1 every time FRINDEX[12] toggles.

**Bit 2 – PCD** Port Change Detect (Cleared on write)
The Host Controller sets this bit to 1 when any port for which the Port Owner bit is set to 0 (see UHPHS Port Status and Control Register) has a change bit transition from 0 to 1 or a Force Port Resume bit transition from 0 to 1 as a result of a J-K transition detected on a suspended port. This bit will also be set as a result of the Connect Status Change being set to 1 after system software has relinquished ownership of a connected port by writing 1 to a port's Port Owner bit.

This bit is allowed to be maintained in the Auxiliary power well. Alternatively, it is also acceptable that on a D3 to D0 transition of the EHCI HC device, this bit is loaded with the OR of all of the PORTSC change bits (including: Force Port Resume, Overcurrent Change, Enable/Disable Change and Connect Status Change).

**Bit 1 – USBERRINT** USB Error Interrupt (Cleared on write)
The Host Controller sets this bit to 1 when completion of a USB transaction results in an error condition (e.g., error counter underflow). If the TD on which the error interrupt occurred also had its IOC bit set, both this bit and USBINT bit are set.

**Bit 0 – USBINT** USB Interrupt (Cleared on write)
The Host Controller sets this bit to 1 on the completion of a USB transaction, which results in the retirement of a Transfer Descriptor that had its IOC bit set.

The Host Controller also sets this bit to 1 when a short packet is detected (the actual number of bytes received was less than the expected number of bytes).

### 41.7.6 UHPHS USB Interrupt Enable Register

**Name:** UHPHS_USBINTR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read/Write

This register enables and disables reporting of the corresponding interrupt to the software. When a bit is set and the corresponding interrupt is active, an interrupt is generated to the host. Interrupt sources that are disabled in this register still appear in the UHPHS_USBSTS to allow the software to poll for events.

For all bits, 1=Enabled, 0=Disabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | IAAE | HSEE | FLRE | PCIE | USBEIE | USBIE |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 5 – IAAE** Interrupt on Async Advance Enable
The interrupt is acknowledged by software clearing the Interrupt on Async Advance bit UHPHS_USBSTS.

**Bit 4 – HSEE** Host System Error Enable
The interrupt is acknowledged by software clearing the Host System Error bit in UHPHS_USBSTS.

**Bit 3 – FLRE** Frame List Rollover Enable
The interrupt is acknowledged by software clearing the Frame List Rollover in UHPHS_USBSTS.

**Bit 2 – PCIE** Port Change Interrupt Enable
The interrupt is acknowledged by software clearing the Port Change Detect bit in UHPHS_USBSTS.

**Bit 1 – USBEIE** USB Error Interrupt Enable
The interrupt is acknowledged by software clearing the USBERRINT in UHPHS_USBSTS.

**Bit 0 – USBIE** USB Interrupt Enable
The interrupt is acknowledged by software clearing the USBINT in UHPHS_USBSTS.

© 2018 Microchip Technology Inc.

### 41.7.7 UHPHS USB Frame Index Register

**Name:** UHPHS_FRINDEX
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read/Write

This register is used by the host controller to index into the periodic frame list. The register updates every 125 µs (once each microframe). Bits [N:3] are used to select a particular entry in the Periodic Frame List during periodic schedule execution. The number of bits used for the index depends on the size of the frame list as set by system software in the Frame List Size field in the UHPHS_USBCMD register).

This register must be written as a DWord. Byte writes produce undefined results. This register cannot be written unless the Host Controller is in the Halted state as indicated by the HCHLT bit (UHPHS_USBSTS register). A write to this register while the Run/Stop bit is set to 1 (UHPHS_USBCMD register) produces undefined results. Writes to this register also affect the SOF value.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | FI[13:8] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FI[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 13:0 – FI[13:0]** Frame Index
The value in this register increments at the end of each time frame (e.g., microframe). Bits [N:3] are used for the Frame List current index. This means that each location of the frame list is accessed eight times (frames or microframes) before moving to the next index. The following illustrates values of N based on the value of FLS (Frame List Size) in the UHPHS_USBCMD register.

| UHPHS_USBCMD.FLS | Number Elements | N |
|---|---|---|
| 0 | 1024 | 12 |
| 1 | 512 | 11 |
| 2 | 256 | 10 |
| 3 | Reserved | – |

The SOF frame number value for the bus SOF token is derived or alternatively managed from this register. The value of FRINDEX must be 125 µs (1 microframe) ahead of the SOF token value. The SOF value may be implemented as an 11-bit shadow register. For this discussion, this shadow register is 11 bits and is named SOFV. SOFV updates every eight microframes (1 millisecond). An example implementation to achieve this behavior is to increment SOFV each time the FRINDEX[2:0] increments from 0 to 1.

Software must use the value of FRINDEX to derive the current microframe number, both for high-speed isochronous scheduling purposes and to provide the “get microframe number” function required for client drivers. Therefore, the value of FRINDEX and the value of SOFV must be kept consistent if chip is reset or software writes to FRINDEX. Writes to FRINDEX must also write-through FRINDEX[13:3] to SOFV[10:0]. In order to keep the update as simple as possible, software should never write a FRINDEX value where the three least significant bits are 7 or 0.

### 41.7.8 UHPHS Periodic Frame List Base Address Register

**Name:** UHPHS_PERIODICLISTBASE
**Offset:** 0x24
**Reset:** 0x00000000
**Property:** Read/Write

This 32-bit register contains the beginning address of the Periodic Frame List in the system memory. If the host controller is in 64-bit mode (as indicated by a 1 in the 64-bit Addressing Capability field in the UHPHS_HCCSPARAMS register), then the most significant 32 bits of every control data structure address comes from the UHPHS_CTRLDSSEGMENT register. System software loads this register prior to starting the schedule execution by the Host Controller. The memory structure referenced by this physical memory pointer is assumed to be 4-Kbyte aligned. The contents of this register are combined with the Frame Index Register (UHPHS_FRINDEX) to enable the Host Controller to step through the Periodic Frame List in sequence. This register must be written as a DWord. Byte writes produce undefined results.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BA[19:12] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BA[11:4] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BA[3:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | | | | |
| Reset | 0 | 0 | 0 | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 31:12 – BA[19:0]** Base Address (Low)
These bits correspond to memory address signals [31:12], respectively.

### 41.7.9 UHPHS Asynchronous List Address Register

**Name:** UHPHS_ASYNCLISTADDR
**Offset:** 0x28
**Reset:** 0x00000000
**Property:** Read/Write

This 32-bit register contains the address of the next asynchronous queue head to be executed. If the host controller is in 64-bit mode (as indicated by a 1 in the 64-bit Addressing Capability field in the UHPHS_HCCPARAMS register), then the most significant 32 bits of every control data structure address comes from the UHPHS_CTRLDSSEGMENT register. Bits [4:0] of this register cannot be modified by system software and will always return a zero when read. The memory structure referenced by this physical memory pointer is assumed to be 32-byte (cache line) aligned. This register must be written as a DWord. Byte writes produce undefined results.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | LPL[26:19] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | LPL[18:11] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LPL[10:3] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LPL[2:0] | | | | | | | |
| Access | R/W | R/W | R/W | | | | | |
| Reset | 0 | 0 | 0 | | | | | |

**Bits 31:5 – LPL[26:0]** Link Pointer Low
These bits correspond to memory address signals [31:5], respectively. This field may only reference a Queue Head (QH).

### 41.7.10 UHPHS Configure Flag Register

**Name:** UHPHS_CONFIGFLAG
**Offset:** 0x50
**Reset:** 0x00000000
**Property:** Read/Write

This register is in the auxiliary power well. It is only reset by hardware when the auxiliary power is initially applied or in response to a host controller reset.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CF |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – CF** Configure Flag
Host software sets this bit as the last action in its process of configuring the Host Controller. This bit controls the default port-routing control logic. Bit values and side-effects are listed below.

| Value | Description |
|---|---|
| 0 | Port routing control logic default-routes each port to an implementation-dependent classic host controller (default value). |
| 1 | Port routing control logic default-routes all ports to this host controller. |

### 41.7.11 UHPHS Port Status and Control Register

**Name:** UHPHS_PORTSCx
**Offset:** 0x54 + x*0x04 [x=0..2]
**Reset:** 0x00002000
**Property:** Read/Write

The number of port registers is documented in the UHPHS_HCSPARAMS register. Software uses this information as an input parameter to determine how many ports need to be serviced. All ports have the structure defined below.

This register is in the auxiliary power well. It is only reset by hardware when the auxiliary power is initially applied or in response to a host controller reset. The initial conditions of a port are:

- No device connected
- Port disabled

If the port has port power control, software cannot change the state of the port until after it applies power to the port by setting port power to a 1. Software must not attempt to change the state of the port until after power is stable on the port. The host is required to have power stable to the port within 20 milliseconds of the 0 to 1 transition.

**Note:**

1. When a device is attached, the port state transitions to the connected state and system software will process this as with any status change notification.
2. If a port is being used as the Debug Port, then the port may report device connected and enabled when the Configured Flag is set to 0.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | WKOC_E | WKDSCNNT_E | WKCNNT_E | PTC[3:0] | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PIC[1:0] | | PO | PP | LS[1:0] | | | PR |
| Access | R/W | R/W | R/W | R-R/W | R | R | | R/W |
| Reset | 0 | 0 | 1 | 0 | 0 | 0 | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SUS | FPR | OCC | OCA | PEDC | PED | CSC | CCS |
| Access | R/W | R/W | R/W | R | R/W | R/W | R/W | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 22 – WKOC_E** Wake on Overcurrent Enable
This field is 0 if Port Power is 0.

| Value | Description |
|---|---|
| 0 | Disables the port to be sensitive to overcurrent conditions as wake-up events. |
| 1 | Enables the port to be sensitive to overcurrent conditions as wake-up events. |

**Bit 21 – WKDSCNNT_E** Wake on Disconnect Enable
This field is 0 if Port Power is 0.

| Value | Description |
|---|---|
| 0 | Disables the port to be sensitive to device disconnects as wake-up events. |
| 1 | Enables the port to be sensitive to device disconnects as wake-up events. |

**Bit 20 – WKCNNT_E** Wake on Connect Enable
This field is 0 if Port Power is 0.

| Value | Description |
|---|---|
| 0 | Disables the port to be sensitive to device connects as wake-up events. |
| 1 | Enables the port to be sensitive to device connects as wake-up events. |

**Bits 19:16 – PTC[3:0]** Port Test Control
When this field is set to 0, the port is NOT operating in a test mode. A non-zero value indicates that it is operating in test mode and the specific test mode is indicated by the specific value.

Test mode bits are encoded as follows (6 to 15 are reserved):

| Value | Test Mode |
|---|---|
| 0 | Test mode not enabled |
| 1 | Test J_STATE |
| 2 | Test K_STATE |
| 3 | Test SE0_NAK |
| 4 | Test Packet |
| 5 | Test FORCE_ENABLE |

Refer to the USB Specification Revision 2.0, Chapter 7, for details on each test mode.

**Bits 15:14 – PIC[1:0]** Port Indicator Control
Writing to these bits has no effect if the P_INDICATOR bit in the UHPHS_HCSPARAMS register is set to 0. If the P_INDICATOR bit is set to 1, then the bits are encoded as follows:

| Value | Meaning |
|---|---|
| 0 | Port indicators are off |
| 1 | Amber |
| 2 | Green |
| 3 | Undefined |

Refer to the USB Specification Revision 2.0 for a description of how these bits are to be used.

This field is 0 if Port Power is 0.

**Bit 13 – PO** Port Owner
System software uses this field to release ownership of the port to a selected host controller (in the event that the attached device is not a high-speed device). Software writes 1 to this bit when the attached device is not a high-speed device. A 1 in this bit means that a companion host controller owns and controls the port.

| Value | Description |
|---|---|
| 0 | This bit unconditionally goes to a 0 when the bit UHPHS_CONFIGFLAG.CF makes a 0 to 1 transition. |
| 1 | This bit unconditionally goes to 1 whenever the bit UHPHS_CONFIGFLAG.CF=0. |

**Bit 12 – PP** Port Power
The function of this bit depends on the value of the Port Power Control (PPC) field in the UHPHS_HCSPARAMS register. When host controller has port power control switches (PPC=0), PP is in read-only mode:

| Value | Description |
|---|---|
| 1 | Each port is hard-wired to power. |

When host controller has port power control switches (PPC=1), PP is in read/write mode:

| Value | Description |
|---|---|
| 0 | Host port power switch is OFF.<br>When power is not available on a port (i.e., PP at 0), the port is non-functional and does not report attaches, detaches, etc. |
| 1 | Host port power switch is ON.<br>When power is not available on a port (i.e., PP at 0), the port is non-functional and does not report attaches, detaches, etc. |

When an overcurrent condition is detected on a powered port and PPC is set to 1, the PP bit in each affected port may be transitioned by the host controller from 1 to 0 (removing power from the port).

**Bits 11:10 – LS[1:0]** Line Status
These bits reflect the current logical levels of the D+ (bit 11) and D- (bit 10) signal lines. These bits are used for detection of low-speed USB devices prior to the port reset and enable sequence. This field is valid only when the port enable bit is 0 and the current connect status bit is set to 1.

This value of this field is undefined if Port Power is 0.

| Value | Name | Description |
|---|---|---|
| 0 | SE0 | Not a low-speed device, perform EHCI reset |
| 1 | K-STATE | Low-speed device, release ownership of port |
| 2 | J-STATE | Not a low-speed device, perform EHCI reset |
| 3 | Undefined | Not a low-speed device, perform EHCI reset |

**Bit 8 – PR** Port Reset
When software writes a 1 to this bit (from 0), the bus reset sequence as defined in the USB Specification Revision 2.0 is started. Software writes a 0 to this bit to terminate the bus reset sequence. Software must

keep this bit set to 1 long enough to ensure the reset sequence, as specified in the USB Specification Revision 2.0, completes.

**Note:** When software writes this bit to 1, it must also write 0 to the Port Enable bit.

When software writes a 0 to this bit, there may be a delay before the bit status changes to 0. The bit status will not read as 0 until after the reset has completed. If the port is in High-Speed mode after reset is complete, the host controller will automatically enable this port (e.g., set the Port Enable bit to 1). A host controller must terminate the reset and stabilize the state of the port within 2 milliseconds of software transitioning this bit from 1 to 0. For example: if the port detects that the attached device is high-speed during reset, then the host controller must have the port in the enabled state within 2 ms of software writing this bit to 0.

The HCHLT bit in the UHPHS_USBSTS register should be set to 0 before software attempts to use this bit. The host controller may hold Port Reset asserted to 1 when the HCHLT bit is 1.

This field is 0 if Port Power is 0.

| Value | Description |
|---|---|
| 0 | Port is not in Reset. |
| 1 | Port is in Reset. |

**Bit 7 – SUS** Suspend

| Value | Description |
|---|---|
| 0 | Port not in suspend state. |
| 1 | Port in suspend state. |

**Note:**
Port Enabled Bit and Suspend bit of this register define the port states as follows:

| Bits [Port Enabled, Suspend] | Port State |
|---|---|
| 0X | Disable |
| 10 | Enable |
| 11 | Suspend |

When in suspend state, downstream propagation of data is blocked on this port, except for port reset. The blocking occurs at the end of the current transaction, if a transaction was in progress when this bit was written to 1. In the suspend state, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB.

A write of 0 to this bit is ignored by the host controller. The host controller will unconditionally set this bit to 0 when:

- Software sets the Force Port Resume bit to 0 (from 1).
- Software sets the Port Reset bit to 1 (from 0).

If host software sets this bit to 1 when the port is not enabled (i.e., Port Enabled bit set to 0), the results are undefined.

This field is 0 if Port Power is set to 0.

**Bit 6 – FPR** Force Port Resume
This functionality defined for manipulating this bit depends on the value of the Suspend bit. For example, if the port is not suspended (Suspend and Enabled bits are set to 1) and software transitions this bit to 1, then the effects on the bus are undefined.

Software sets this bit to a 1 to drive resume signaling. The Host Controller sets this bit to 1 if a J-to-K transition is detected while the port is in the Suspend state. When this bit transitions to 1 because a J-to-K transition is detected, the Port Change Detect bit in the UHPHS_USBSTS register is also set to 1. If software sets this bit to 1, the host controller must not set the Port Change Detect bit.

Note that when the EHCI controller owns the port, the resume sequence follows the defined sequence documented in the USB Specification Revision 2.0. The resume signaling (Full-speed 'K') is driven on the port as long as this bit remains set to 1. Software must appropriately time the Resume and set this bit to 0 when the appropriate amount of time has elapsed. Writing a 0 (from 1) causes the port to return to High-Speed mode (forcing the bus below the port into a high-speed idle). This bit will remain set to 1 until the port has switched to the high-speed idle. The host controller must complete this transition within 2 milliseconds of software setting this bit to 0.

This field is 0 if Port Power is 0.

| Value | Description |
|---|---|
| 0 | No resume (K-state) detected/driven on port. |
| 1 | Resume detected/driven on port. |

**Bit 5 – OCC** Overcurrent Change (Cleared on write)
Software clears this bit by writing 1.

| Value | Description |
|---|---|
| 0 | No change to Overcurrent Active. |
| 1 | Changes to Overcurrent Active. |

**Bit 4 – OCA** Overcurrent Active
This bit will automatically transition from 1 to 0 when the overcurrent condition is removed.

| Value | Description |
|---|---|
| 0 | This port does not have an overcurrent condition. |
| 1 | This port currently has an overcurrent condition. |

**Bit 3 – PEDC** Port Enable/Disable Change (Cleared on write)
For the root hub, this bit gets set to 1 only when a port is disabled due to the appropriate conditions existing at the EOF2 point (See Chapter 11 of the USB Specification for the definition of a Port Error). Software clears this bit by writing a 1 to it.

This field is 0 if Port Power bit is 0.

| Value | Description |
|---|---|
| 0 | No change in port enabled/disabled status. |
| 1 | Port enabled/disabled status has changed. |

**Bit 2 – PED** Port Enabled/Disabled
Ports can only be enabled by the host controller as a part of the reset and enable. Software cannot enable a port by writing a 1 to this field. The host controller will only set this bit to 1 when the reset sequence determines that the attached device is a high-speed device.

Ports can be disabled by either a fault condition (disconnect event or other fault condition) or by host software. Note that the bit status does not change until the port state actually changes. There may be a delay in disabling or enabling a port due to other host controller and bus events.

When the port is disabled (0b), downstream propagation of data is blocked on this port, except for reset.

This field is 0 if Port Power bit is 0.

| Value | Description |
|---|---|
| 0 | Disable. |
| 1 | Enable. |

**Bit 1 – CSC** Connect Status Change (Cleared on write)
Indicates a change has occurred in the port's Current Connect Status. The host controller sets this bit for all changes to the port device connect status, even if system software has not cleared an existing connect status change. For example, the insertion status changes twice before system software has cleared the changed condition, hub hardware will be "setting" an already-set bit (i.e., the bit remains set). Software sets this bit to 0 by writing a 1 to it.

This field is 0 if Port Power bit is 0.

| Value | Description |
|---|---|
| 0 | No change. |
| 1 | Change in Current Connect Status. |

**Bit 0 – CCS** Current Connect Status
This value reflects the current state of the port, and may not correspond directly to the event that caused the CSC bit to be 1.

This bit is 0 if Port Power is 0.

| Value | Description |
|---|---|
| 0 | No device is present. |
| 1 | Device is present on port. |

### 41.7.12 EHCI: REG06 - AHB Error Status

**Name:** UHPHS_INSNREG06
**Offset:** 0xA8
**Reset:** 0x00000000
**Property:** Read/Write

Control and Status Register, used to read the UTMI registers from the signals below.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | AHB_ERR | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | HBURST[2:0] | | | Nb_Burst[4:4] |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | Nb_Burst[3:0] | | | | Nb_Success_Burst[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – AHB_ERR** AHB Error
System bus error was encountered and erroneous burst characteristics are captured. To clear this field the application must write a 0.

EHCI:

- When no error, 0 is written to INSNREG06[8:4].
- When INCR4 and an error occurs, 4 is written to INSNREG06[8:4].
- When INCR8 and an error occurs, 8 is written to INSNREG06[8:4].
- When INCR16 and an error occurs, 16 is written to INSNREG06[8:4].
- Other values except 4, 8, and 16 are not written to INSNREG06[8:4].

OHCI:

- When no error, 0 is written to INSNREG06[8:4].
- When INCR4 and error occurs, 4 is written to INSNREG06[8:4].
- Other values except 4 are not written to INSNREG06[8:4].

**Bits 11:9 – HBURST[2:0]** Burst Value
Value of the control phase at which the AHB error occurred.

This field applies to enabled incremental bursts only.

**Bits 8:4 – Nb_Burst[4:0]** Number of Bursts
Number of beats expected in the burst at which the AHB error occurred. Valid values are 0 to 16.

This field applies to enabled incremental bursts only.

**Bits 3:0 – Nb_Success_Burst[3:0]** Number of Successful Bursts
Number of successfully completed beats in the current burst before the AHB error occurred.

This field applies to enabled incremental bursts only.

### 41.7.13 EHCI: REG07 - AHB Master Error Address

**Name:** UHPHS_INSNREG07
**Offset:** 0xAC
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | AHB_ADDR[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | AHB_ADDR[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | AHB_ADDR[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | AHB_ADDR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – AHB_ADDR[31:0]** AHB Address
System bus address of the control phase at which the system bus error occurred.

### 41.7.14 EHCI: REG08 - HSIC Enable/Disable

**Name:** UHPHS_INSNREG08
**Offset:** 0xB0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | HSIC_EN | | |
| Access | | | | | | R/W | | |
| Reset | | | | | | 0 | | |

**Bit 2 – HSIC_EN** HSIC Enable/Disable
This register has R/W access to the host driver and gives control to the host driver to enable/disable the HSIC interface of Port C.

| Value | Description |
|---|---|
| 0 | Port C is in the HSIC Disable state (see High-Speed Inter-Chip USB Electrical Specification, Version 1.0, Section 3.1.2). HSIC is in the Disabled state after a power-on reset. |
| 1 | Port C is in the HSIC Enable state (see High-Speed Inter-Chip USB Electrical Specification, Version 1.0, Section 3.1.2). |

# 42. Audio Class D Amplifier (CLASSD)

## 42.1 Description

The Audio Class D Amplifier (CLASSD) is a digital input, Pulse Width Modulated (PWM) output stereo Class D amplifier. It features a high quality interpolation filter embedding a digitally controlled gain, an equalizer and a de-emphasis filter.

On its input side, the CLASSD is compatible with most common audio data rates. On the output side, its PWM output can drive either:

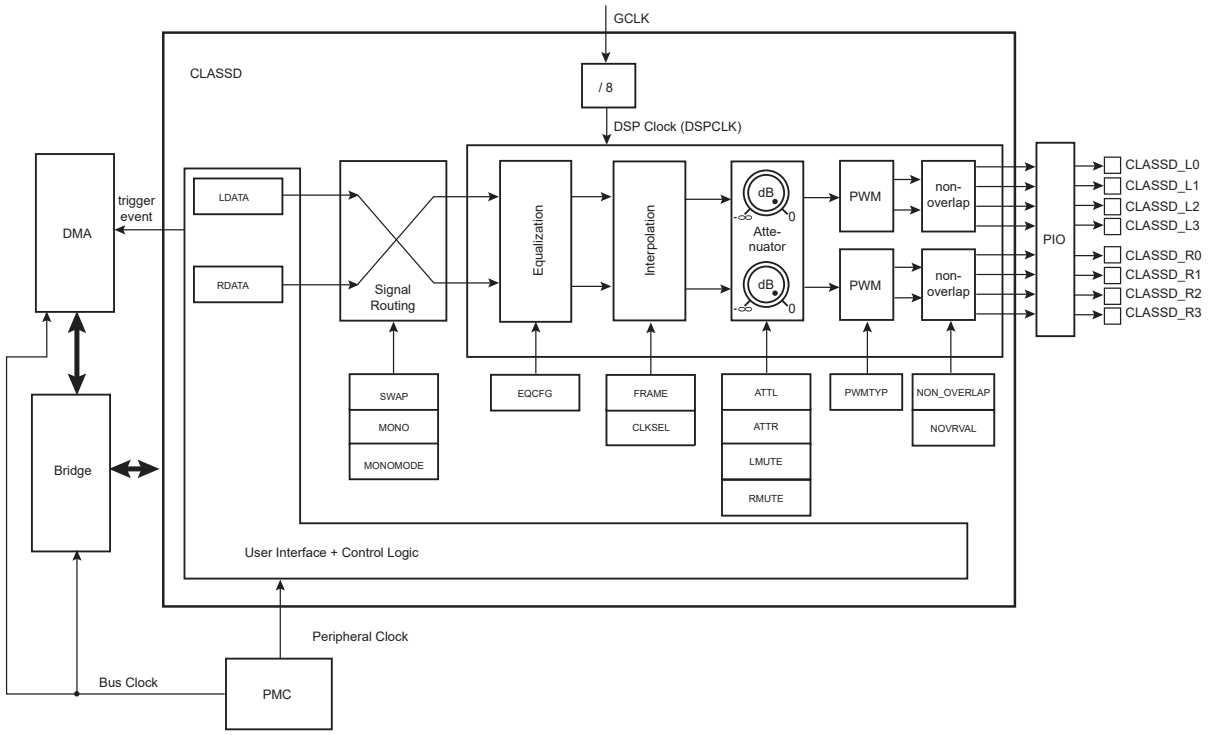
- high-impedance single-ended or differential output loads (Audio DAC application) or,
- external MOSFETs through an integrated non-overlapping circuit (Class D power amplifier application).

## 42.2 Embedded Characteristics

- Stereo PWM Class D Amplifier
- 16-bit Audio Data
- DSP clocks: 12.288 and 11.2896 MHz
- Input Sampling Rates: 8, 16, 32, 48, 96, 22.05, 44.1, 88.2 kHz
- 3-band Equalizer
- De-emphasis Filter
- Digital Volume Control
- Differential or Single-ended Outputs
- Non-overlapping Circuit to Control External MOSFETs
- Supports DMA

## 42.3 Block Diagram

**Figure 42-1. CLASSD Block Diagram**




## 42.4 Pin Name List

**Table 42-1. Output Pins Assignment Versus Application Use Cases**

| Pin | External MOS Driver (NON_OVERLAP = 1) | | Direct Load (NON_OVERLAP = 0) | | Type |
|---|---|---|---|---|---|
| | Full H-Bridge (PWMTYP = 1) | Half H-Bridge (PWMTYP = 0) | Differential Load (PWMTYP = 1) | Single-Ended Load (PWMTYP = 0) | |
| | Use Case 1 | Use Case 2 | Use Case 3A & 3B | Use Case 4A & 4B | |
| CLASSD_L0 | gate_pmos_leftp | gate_pmos_left | leftp | left | Output |
| CLASSD_L1 | gate_nmos_leftp | gate_nmos_left | Not used (fixed to 0) | Not used (fixed to 0) | Output |
| CLASSD_L2 | gate_pmos_leftn | Not used (fixed to 1) | leftn | Not used (fixed to 0) | Output |
| CLASSD_L3 | gate_nmos_leftn | Not used (fixed to 1) | Not used (fixed to 0) | Not used (fixed to 0) | Output |
| CLASSD_R0 | gate_pmos_rightp | gate_pmos_right | rightp | right | Output |

.....continued

| Pin | External MOS Driver (NON_OVERLAP = 1) | | Direct Load (NON_OVERLAP = 0) | | Type |
|---|---|---|---|---|---|
| | Full H-Bridge (PWMTYP = 1) | Half H-Bridge (PWMTYP = 0) | Differential Load (PWMTYP = 1) | Single-Ended Load (PWMTYP = 0) | |
| | Use Case 1 | Use Case 2 | Use Case 3A & 3B | Use Case 4A & 4B | |
| CLASSD_R1 | gate_nmos_rightp | gate_nmos_right | Not used (fixed to 0) | Not used (fixed to 0) | Output |
| CLASSD_R2 | gate_pmos_rightn | Not used (fixed to 1) | rightn | Not used (fixed to 0) | Output |
| CLASSD_R3 | gate_nmos_rightn | Not used (fixed to 1) | Not used (fixed to 0) | Not used (fixed to 0) | Output |

## 42.5 Product Dependencies

### 42.5.1 I/O Lines

The pins used for interfacing the compliant external devices may be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the CLASSD pins to their peripheral functions.

### 42.5.2 Power Management

The CLASSD is clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the CLASSD Peripheral Clock and provide a generic clock (GCLK).

The fields NOVRVAL, NON_OVERLAP and PWMTYP in CLASSD_MR, and DSPCLKFREQ and FREQ in CLASSD_INTPMR, must be configured prior to applying the GCLK.

### 42.5.3 Interrupt

The CLASSD has an interrupt line connected to the interrupt controller. Handling the CLASSD interrupt requires programming the interrupt controller before configuring the CLASSD.

## 42.6 Functional Description

### 42.6.1 Interpolator

#### 42.6.1.1 Clock Configuration

The interpolator accepts input sampling frequencies ($f_s$) and the input DSP clock (DSPCLK) that can be configured in the CLASSD Interpolator Mode Register. GCLK must be configured in the PMC according to the desired DSPCLK so that DSPCLK = GCLK / 8.

The following table provides authorized DSPCLK / $f_s$ ratios and associated filter types.

**Table 42-2. Authorized DSPCLK / $f_s$ Ratios & Filter Types**

| $f_s$ | DSPCLK | |
|---|---|---|
| | 12.288 MHz | 11.2896 MHz |
| 8 kHz | 2 | – |
| 16 kHz | 2 | – |
| 32 kHz | 2 | – |
| 48 kHz | 1 | – |
| 96 kHz | 3 | – |
| 22.05 kHz | – | 1 |
| 44.1 kHz | – | 1 |
| 88.2 kHz | – | 3 |

**Note:** Each dash (–) indicates a configuration that is not authorized and that raises the CFGERR flag in the CLASSD Interpolator Status Register.

#### 42.6.1.2 CLASSD Frequency Response

Interpolation is performed with a combination of Infinite Impulse Response (IIR) and Cascaded Integrator-Comb (CIC) filters. Given the input configuration, the coefficients of the filters are redefined to optimize their transfer function to optimize the audio bandwidth. The different types of filters are defined in section Clock Configuration.

**Figure 42-2. Type 1 Frequency Response**

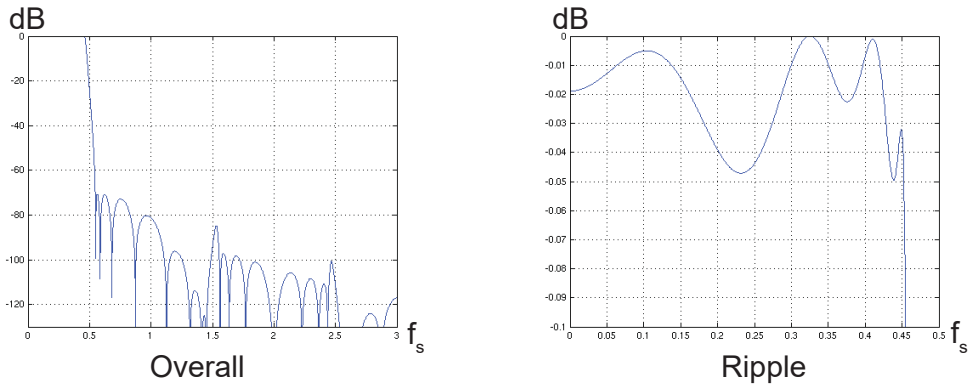


**Figure 42-3. Type 2 Frequency Response**

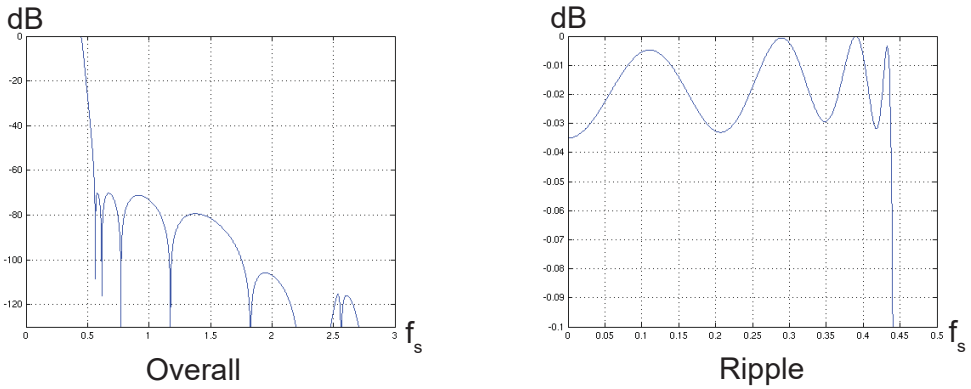
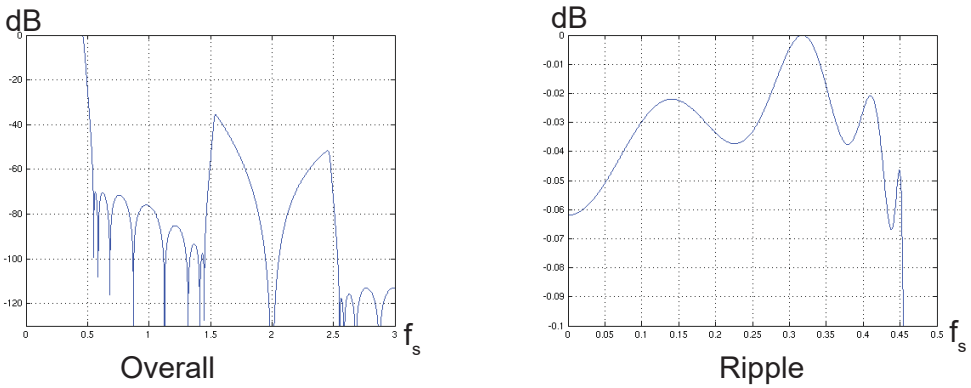



**Figure 42-4. Type 3 Frequency Response**




### 42.6.2 Equalizer

The CLASSD offers 12 pre-programmed equalization filters.

A zero-cross detection system is used to modify the equalizer on-the-fly with minimum disturbance on the output signal.

Programming of the equalization filter is detailed in section CLASSD Interpolator Mode Register.

The following figures show the frequency response of the equalizer function implemented in the D/A channels.

**Figure 42-5. Bass Filters Response**

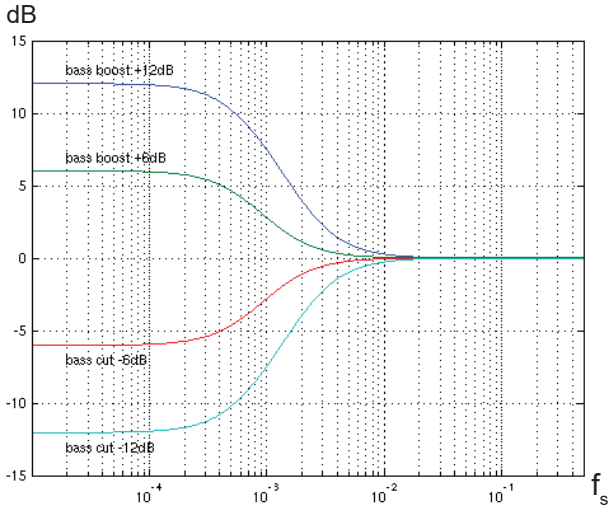



**Figure 42-6. Medium Filters Response**

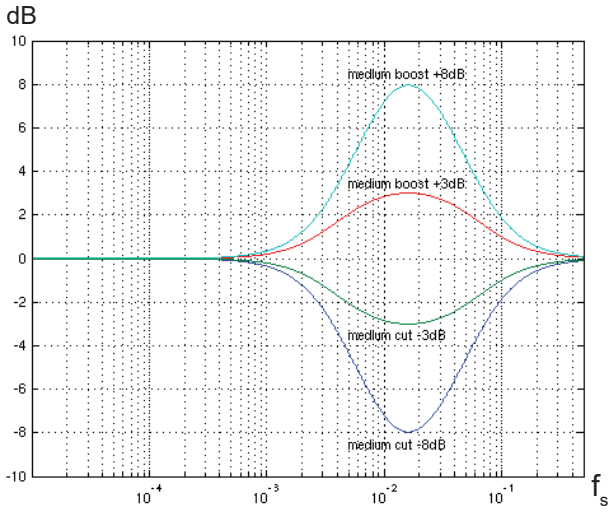
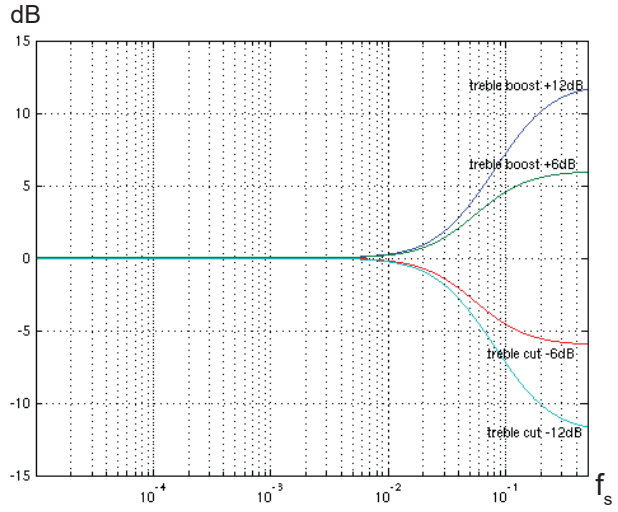


**Figure 42-7. Treble Filters Response**




### 42.6.3 De-emphasis Filter Frequency Response

The CLASSD includes a de-emphasis filter which can be enabled for 32, 44.1 or 48 kHz sampling frequencies.

The response and the error generated by the digital approximation of the filter are illustrated in the following figures.

**Figure 42-8. De-emphasis Filter: Frequency Response & Error ($f_s$ = 32 kHz)**

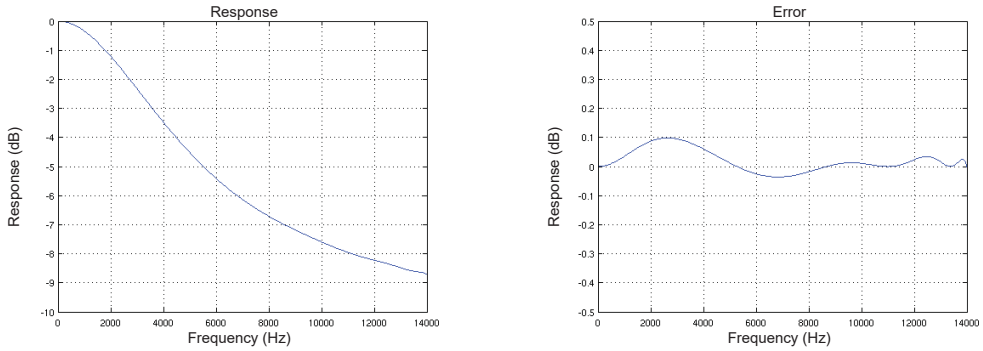



**Figure 42-9. De-emphasis Filter: Frequency Response & Error ($f_s$ = 44.1 kHz)**

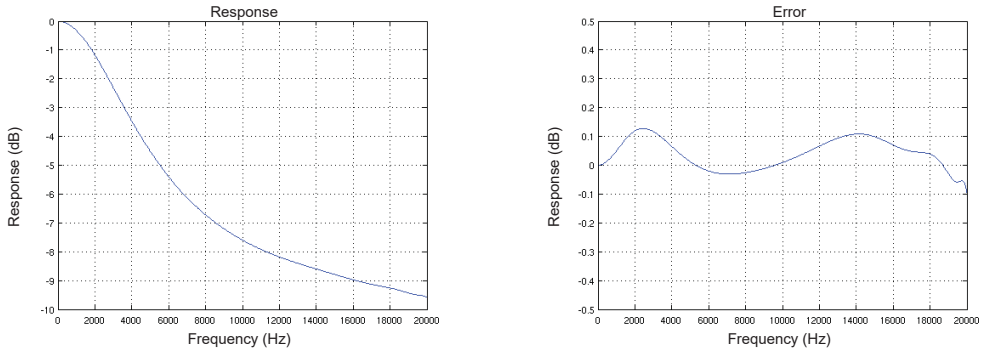
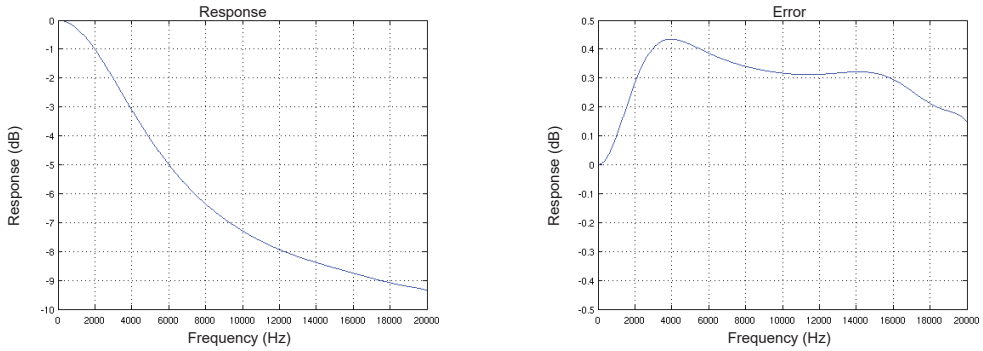


**Figure 42-10. De-emphasis Filter: Frequency Response & Error ($f_s$ = 48 kHz)**




### 42.6.4 Attenuator and Recommended Input Levels

The CLASSD features a digital attenuator with an attenuation range of 0–77 dB and a step size of 1 dB. When attenuation greater than 77 dB is programmed, the attenuator mutes the channel.

To avoid saturations in the PWM stage, it is recommended to avoid input levels greater than 1 dB below the digital full scale (-1 dBFS). This can done by programming a minimum attenuation of 1 dB.

### 42.6.5 Pulse Width Modulator (PWM)

The CLASSD Pulse Width Modulator generates fixed frequency pulse width modulated output signals. For the 44.1 kS/s and 48 kS/s standard audio sample rates, the PWM output frequency is set to 16 × $f_S$: 705.6 kHz and 768 kHz respectively. For 8, 16, 24 and 96 kS/s, the 16× (interpolation) ratio is adapted to keep the output frequency at 768 kHz. In the same way, the output frequency is 705.6 kHz for the 22.05 and 88.2 kS/s cases.

The CLASSD functions either as a DAC loaded by a medium-to-high resistive load (e.g., 1 kΩ to 100 kΩ) or as a Class D power amplifier controller driving an external power stage. Depending on the value of CLASSD_MR.NON_OVERLAP, the CLASSD drives:

- Single-ended or differential resistive loads (NON_OVERLAP = 0)
- Full or Half MOSFET H-bridges (NON_OVERLAP = 1)

When driving an external power stage (NON_OVERLAP = 1), the CLASSD generates the signals to control complementary MOSFET pairs (PMOS and NMOS) with a non-overlapping delay between the NMOS and PMOS controls to avoid short circuit current. The non-overlapping delay can be adjusted in the CLASSD_MR.NOVRVAL field.

The CLASSD can have a single-ended or a differential output. A specific pulse width modulation type is associated to each case. For single-ended output (CLASSD_MR.PWMTYP = 0), the PWM acts only on the falling edge of the PWM waveform (trailing edge PWM). For differential output (CLASSD_MR.PWMTYP = 1), both the rising and the falling edges of the PWM waveform are modulated (symmetric PWM). Modulation principles are illustrated in the following figures for both types of PWM. In particular, when describing a null input, if PWMTYP = 0 (trailing edge PWM), the output waveform is a square wave with 50% duty cycle. With the same input and PWMTYP = 1, the differential output waveform is zero. This difference removes the classical L-C low pass filter when PWMTYP = 1.

**Figure 42-11. Output Waveform Modulation Principle for PWMTYP = 0**

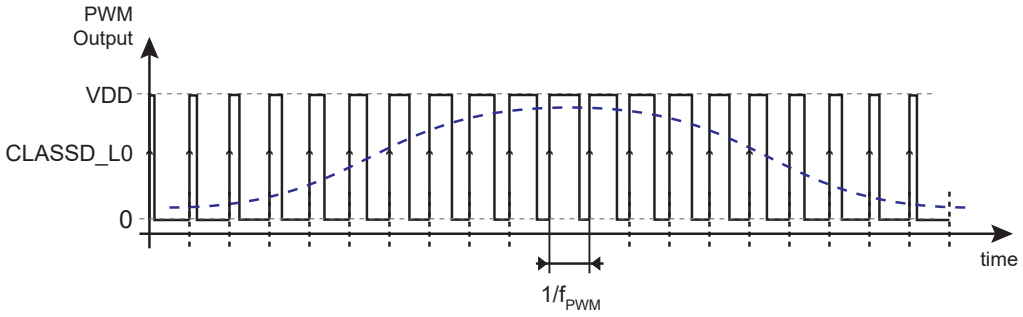
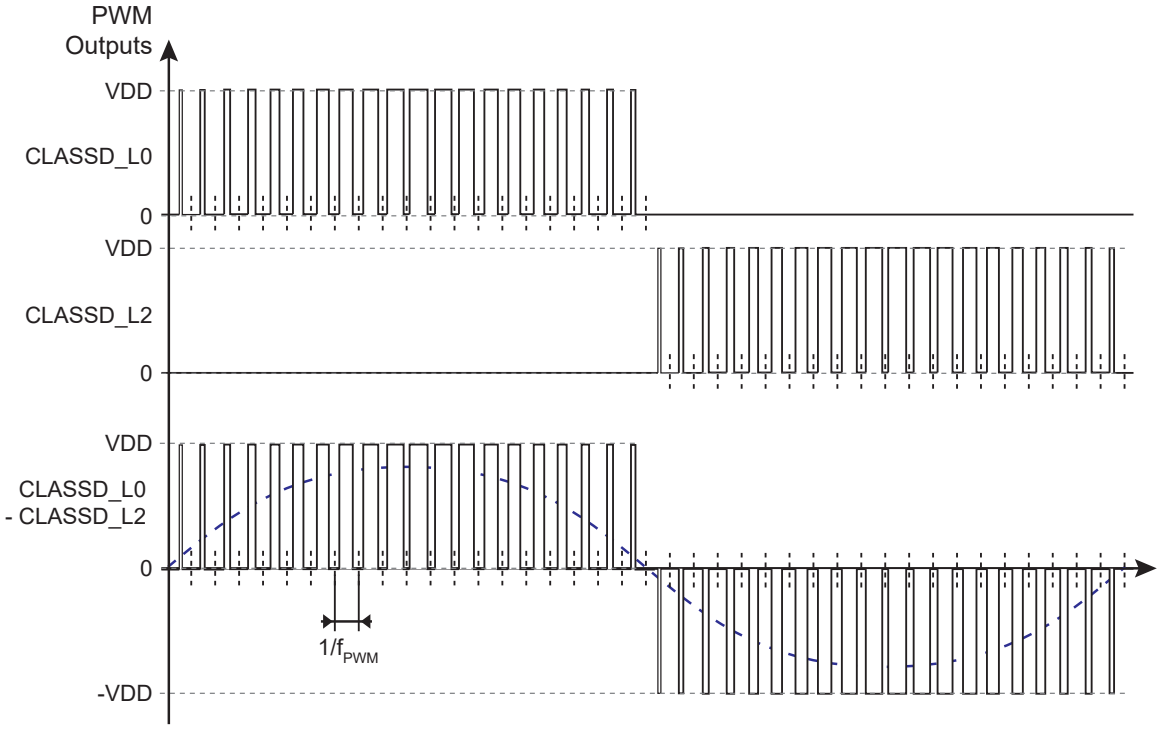
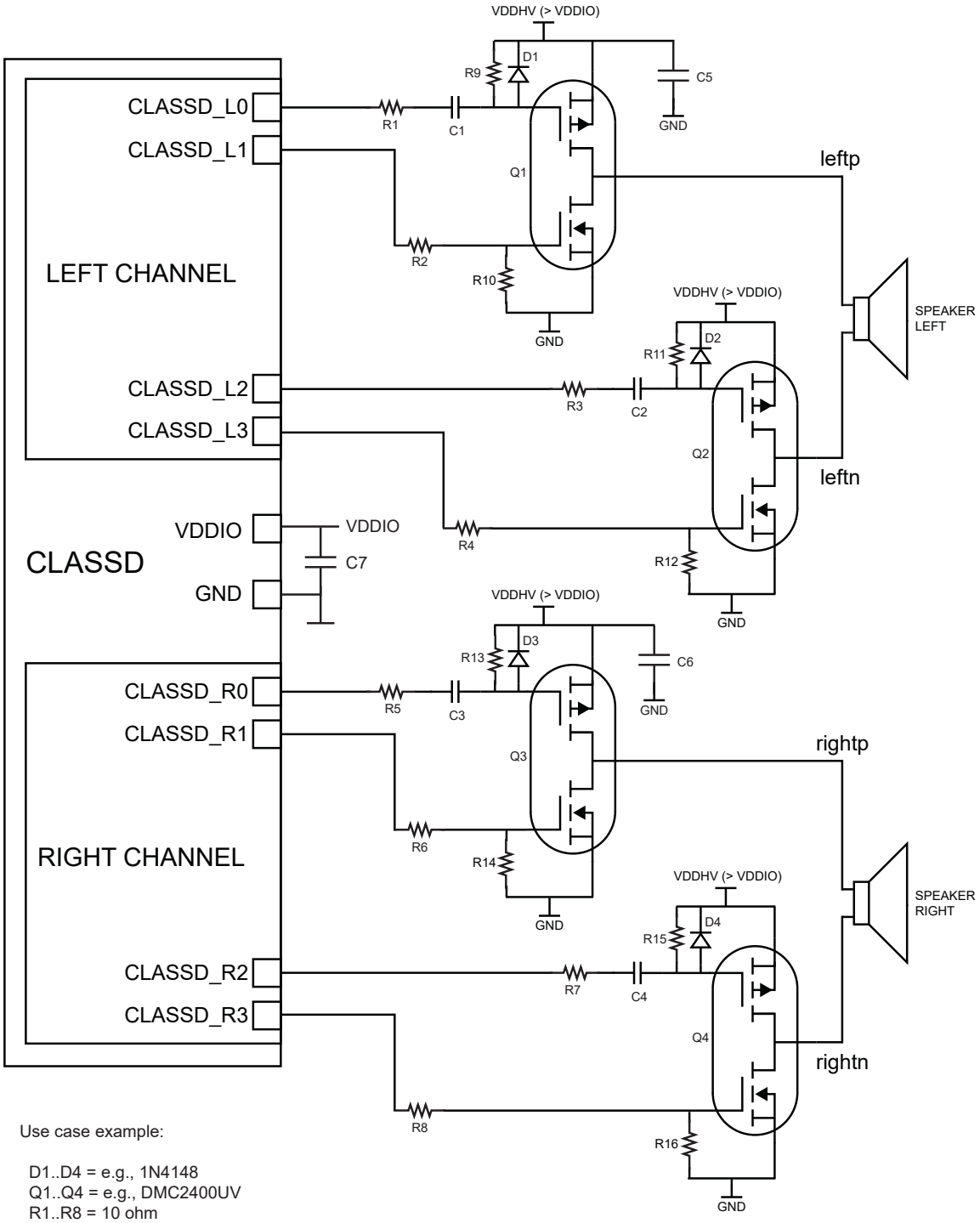



**Figure 42-12. Output Waveform Modulation Principle for PWMTYP = 1 (Only Left Channel Pins Shown)**



### 42.6.6 Application Schematics For Use Case Examples

**Figure 42-13. Use Case 1: Stereo Class D Amplifier With External Differential Power Stage**

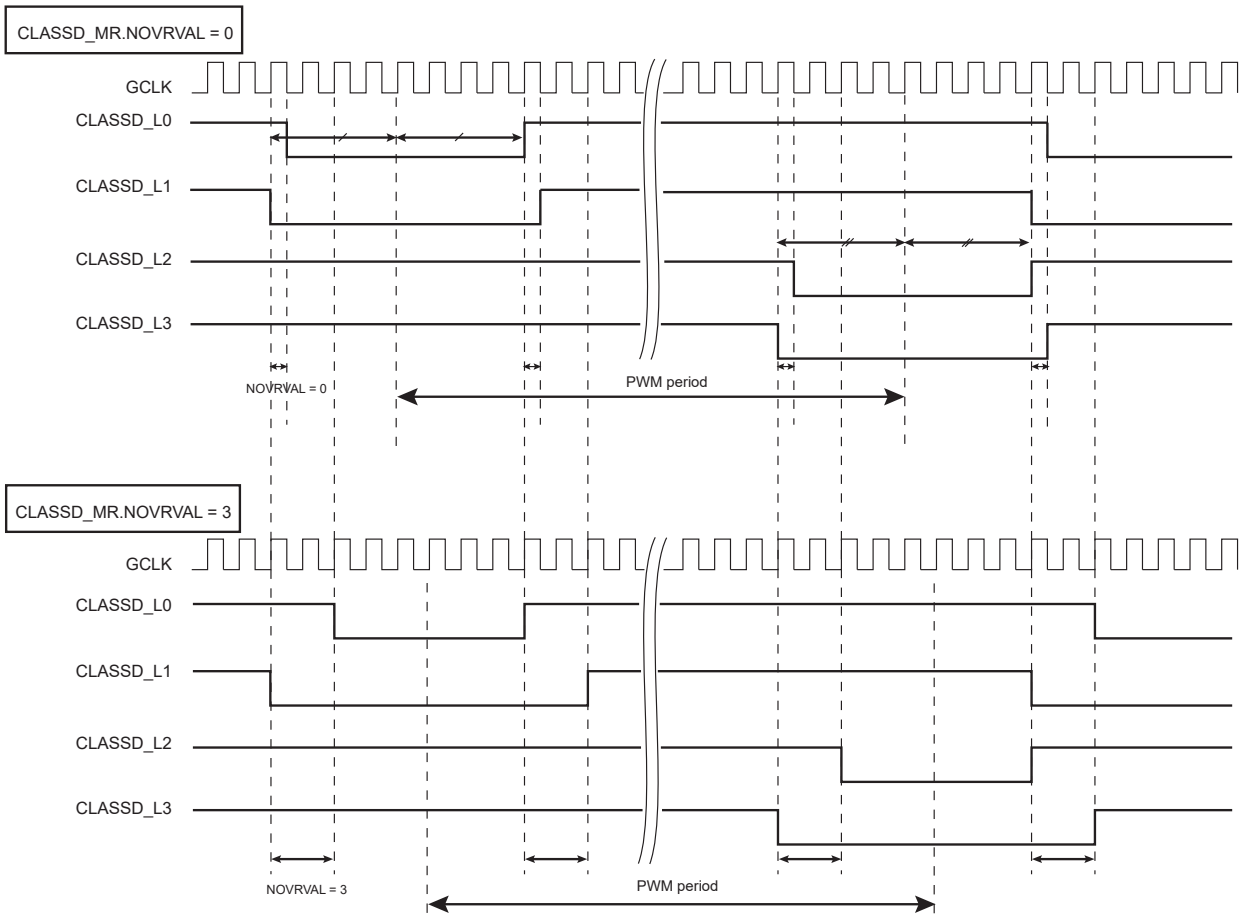



Use case example:

D1..D4 = e.g., 1N4148
Q1..Q4 = e.g., DMC2400UV
R1..R8 = 10 ohm
R9..R16 = 10 kohm
C1..C4 = 10 nF
C5..C6 = 10 µF
C7 = 1 µF

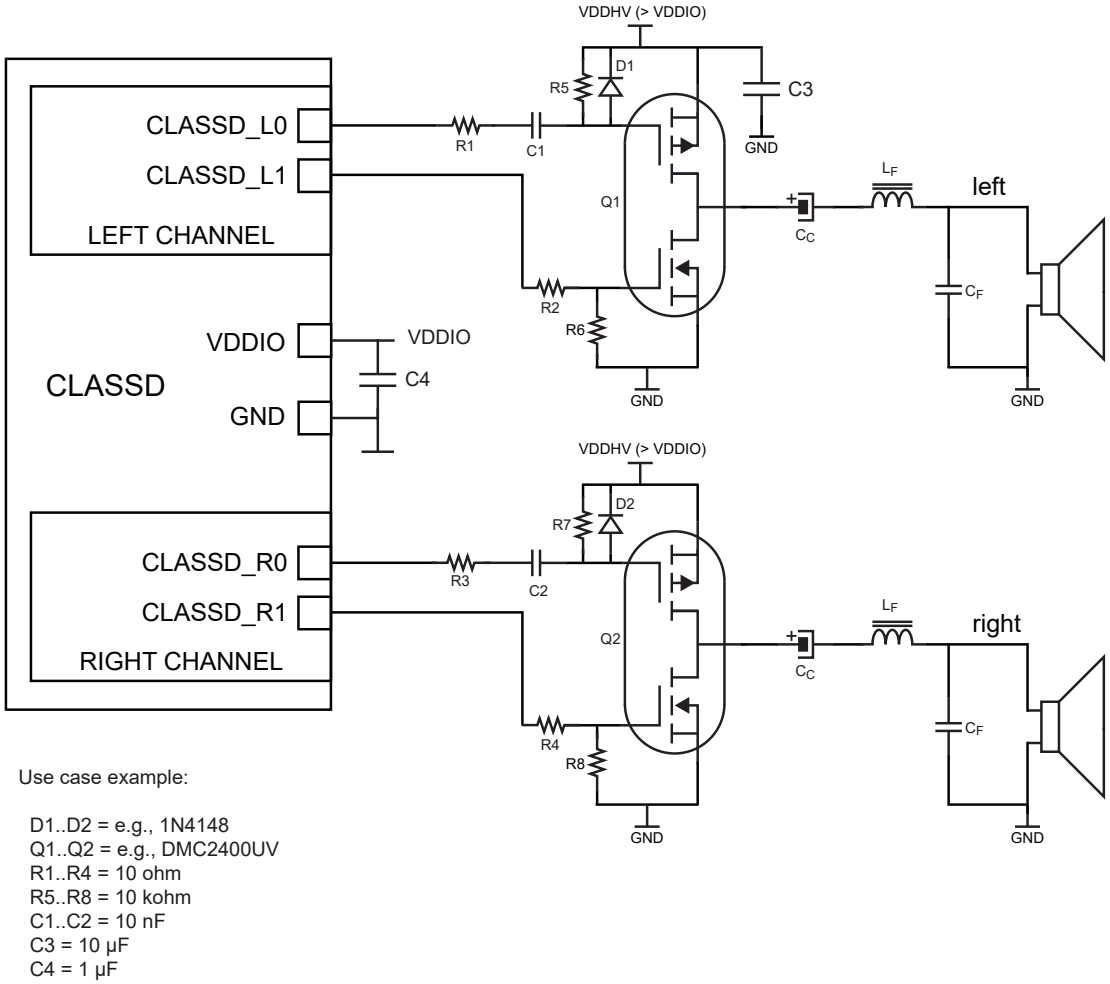
**Figure 42-14. Use Case 1: Waveforms**

CLASSD_MR.PWMTYP = 1, CLASSD_MR.NON_OVERLAP = 1




In Use Case 1, the external power stages are made of complementary low-cost MOSFETs. In addition to the $R_{DSON}$ and drain breakdown voltage characteristics, the choice of these components is driven by a low gate threshold voltage, a low input capacitance, a low total gate charge and a fast turn-on time characteristics. Series resistance (10 Ω) added to the gates of the MOSFETs are optional and may be adjusted to optimize the gate drive. They help to limit the output current peaks driven by the I/Os into the MOSFET gates in some cases. The 10k resistors ensure an OFF condition when not driven and the capacitor / diode network (C1..C2 / D1..D2) shifts the PMOS drive from the typical $V_{DDIO}$ level (3.3V) to a higher supply voltage (e.g., a 5V power domain).

**Figure 42-15. Use Case 2: Stereo Class D Amplifier With External Single-ended Power Stage**



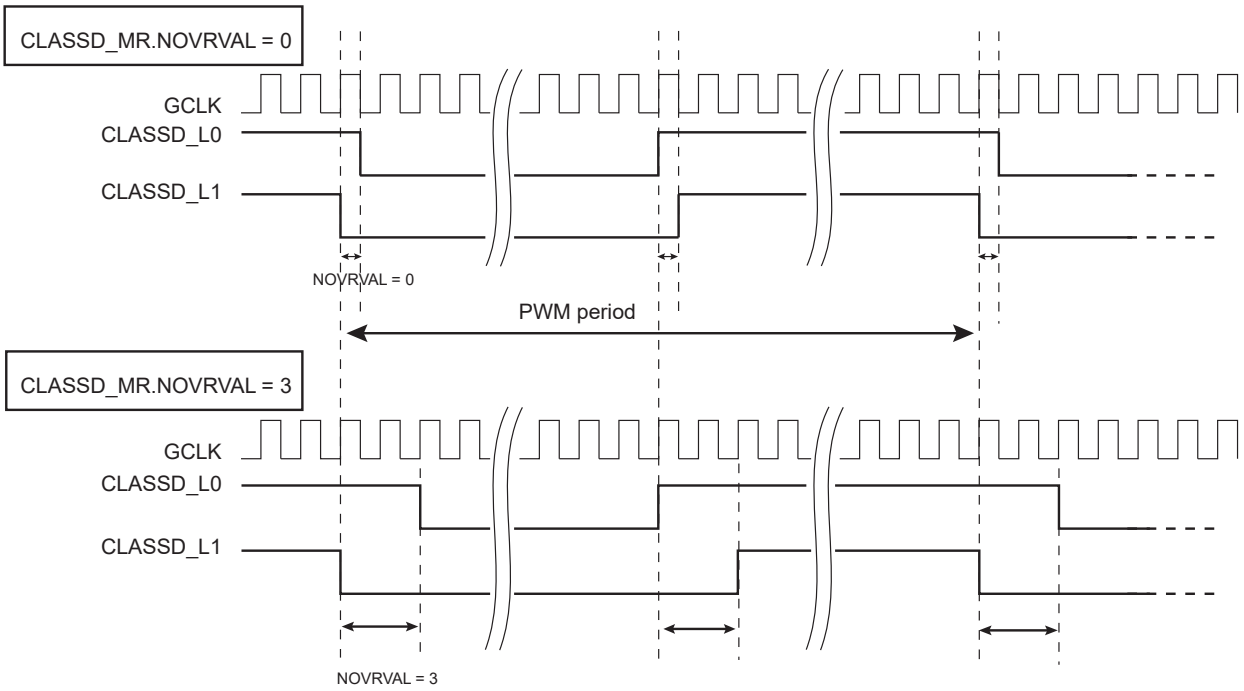

Use case example:

D1..D2 = e.g., 1N4148
Q1..Q2 = e.g., DMC2400UV
R1..R4 = 10 ohm
R5..R8 = 10 kohm
C1..C2 = 10 nF
C3 = 10 µF
C4 = 1 µF

In the Use Case 2 application schematic, the drive network of the MOSFETs gates follows the principles described in Use Case 1.

**Figure 42-16. Use Case 2: Waveforms**

CLASSD_MR.PWMTYP = 0, CLASSD_MR.NON_OVERLAP = 1




A coupling capacitor ($C_C$) and an L-C low pass filter ($L_F$, $C_F$) are added to the output of the power stage to remove both the DC and the high frequency components of the PWM signal. $C_C$ with the resistive part of the speaker ($R_{SPK}$) forms a C-R high pass filter with a corner frequency of $f_{HP} = 1 / (2 \times PI \times C_C \times R_{SPK})$.

$L_F$, $C_F$ and $R_{SPK}$ form a second-order low pass filter of corner frequency $f_C = 1 / (2 \times PI \times sqrt (L_F \times C_F))$ and of quality factor $Q = R_{SPK} \times sqrt (C_F / L_F)$. As a numerical example, consider the case $f_{HP}$ = 200 Hz, $f_C$ = 30 kHz, Q = 0.707 (maximum flat response) with $R_{SPK}$ = 8 Ω. This leads to $C_C$ = 100 µF, $L_F$ = 60 µH, $C_F$ = 470 nF.

**Figure 42-17. Use Case 3A: Stereo Audio DAC With Active differential to single low pass filter**

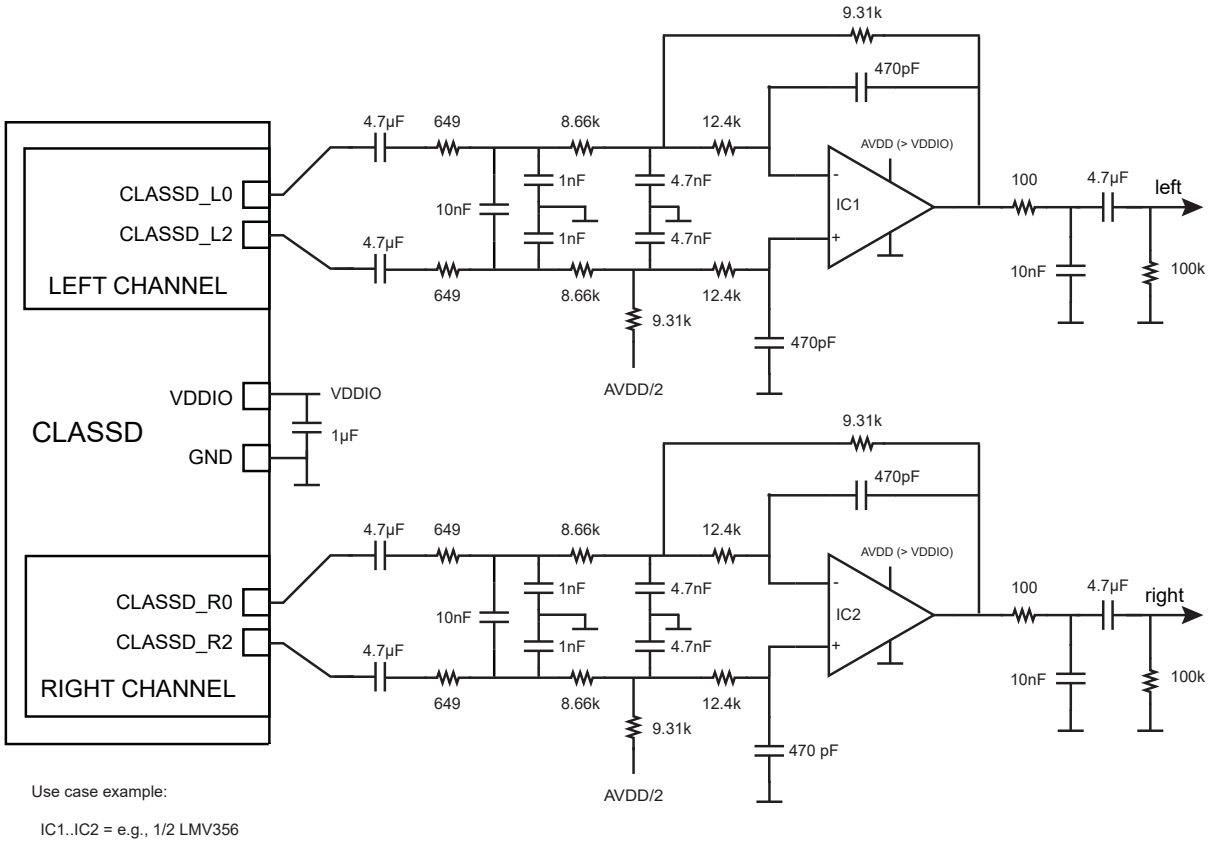



**Figure 42-18. Use Case 3B: Stereo Audio DAC With Simple Passive Low Pass Filter and Differential Outputs**

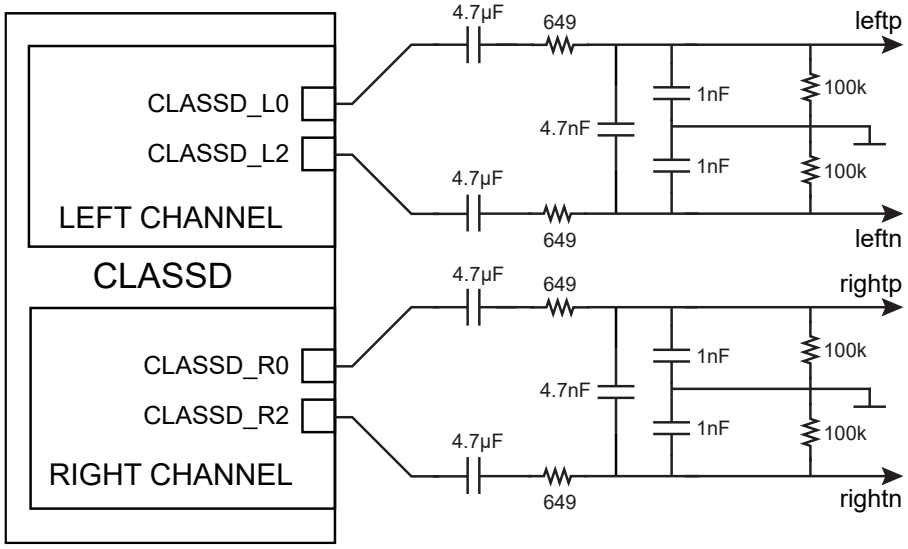
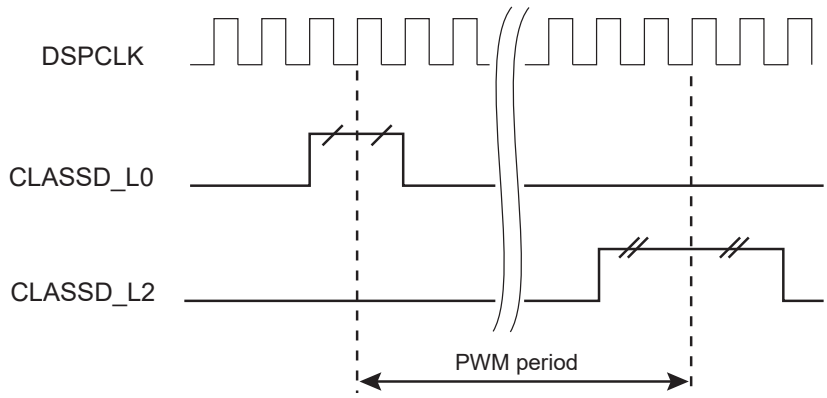


**Figure 42-19. Use Cases 3A and 3B: Waveforms**

CLASSD_MR.PWMTYP = 1, CLASSD_MR.NON_OVERLAP = 0




In Use Case 3A, the CLASSD is used as an audio DAC. In this case, the differential outputs of the CLASSD are used. The application schematic suggested in figure "Use Case 3A: Stereo Audio DAC With Active differential to single low pass filter" above implements a third order 10 kHz low pass Butterworth filter and makes the differential to single-ended conversion. Note that in this schematic, the AVDD/2 point needs to be fed at low impedance (e.g., a buffered voltage). A simpler schematic (Use Case 3B) may also be possible, as shown in figure "Use Case 3B: Stereo Audio DAC With Simple Passive Low Pass Filter and Differential Outputs" above, at the cost of higher out-of-band noise and differential outputs which may be acceptable in some applications.

**Figure 42-20. Use Case 4A: Stereo Audio DAC With Active Low Pass Filter and Single-ended Outputs**

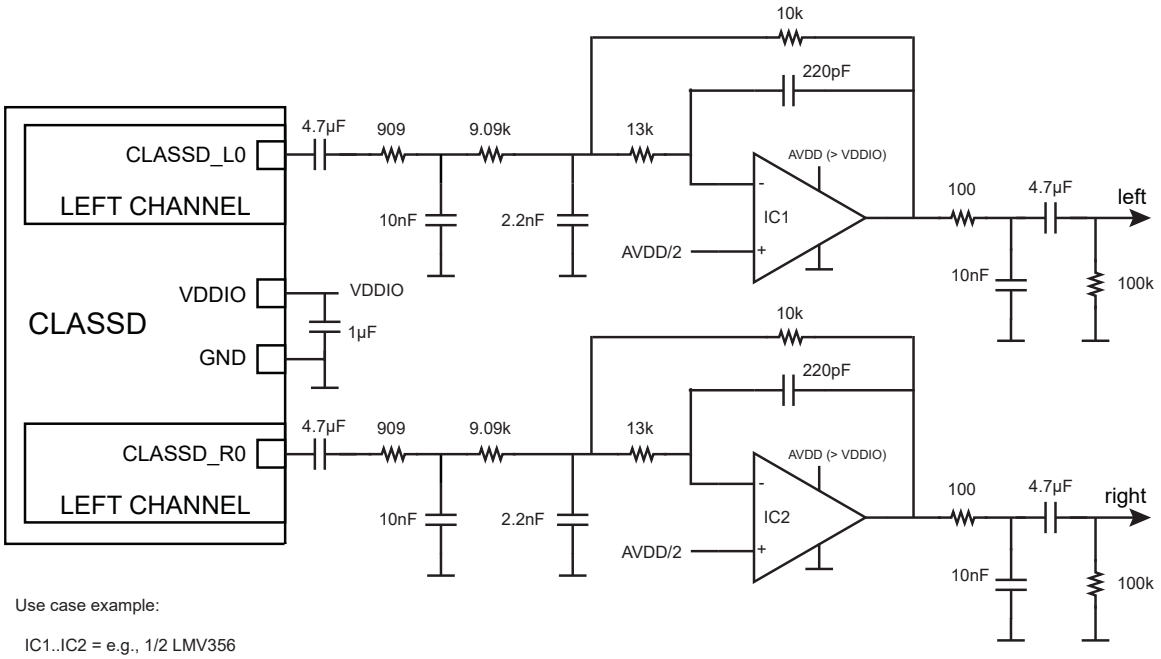


**Figure 42-21. Use Case 4B: Stereo Audio DAC With Passive Low Pass Filter and Single-ended Outputs**

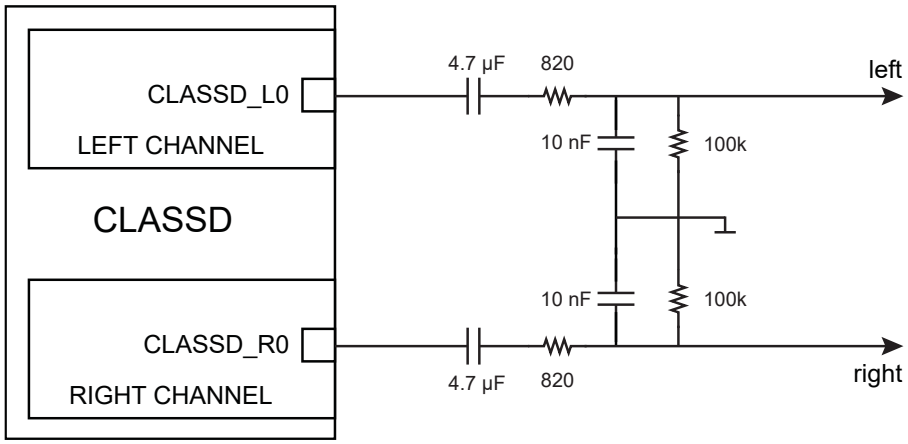
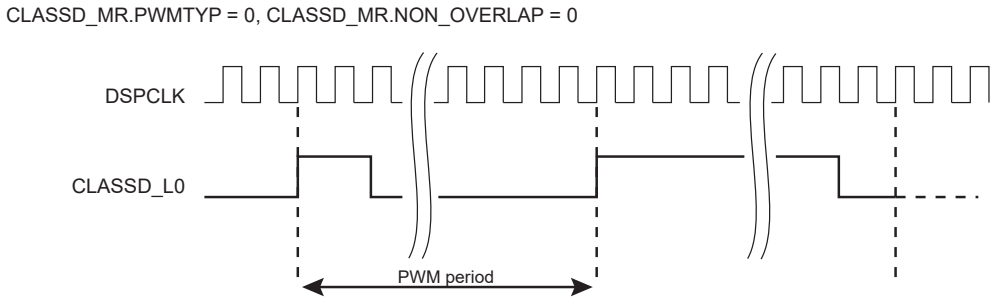



**Figure 42-22. Use Cases 4A and 4B: Waveforms**




In Use Case 4A, the CLASSD is used as an audio DAC with active low pass filter. In this case, the single-ended outputs of the CLASSD are selected (PWMTYP = 0, trailing edge PWM) which leaves more I/Os to the application. A third order 30 kHz low pass Butterworth filter is shown in figure "Use Case 4A: Stereo Audio DAC With Active Low Pass Filter and Single-ended Outputs". The AVDD/2 point can be fed at relatively high impedance as no current is drawn from this point (a simple resistive divider properly decoupled is acceptable). A reduced complexity schematic is presented in figure "Use Case 4B: Stereo Audio DAC With Passive Low Pass Filter and Single-ended Outputs" above for less constrained applications.

### 42.6.7 Register Write Protection

To prevent any single software error from corrupting CLASSD behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the CLASSD Write Protection Mode Register (CLASSD_WPMR).

The following registers can be write-protected:

- CLASSD Mode Register
- CLASSD Interpolator Mode Register

## 42.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CLASSD_CR | 7:0 | | | | | | | | SWRST |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | CLASSD_MR | 7:0 | | | RMUTE | REN | | | LMUTE | LEN |
| | | 15:8 | | | | | | | | PWMTYP |
| | | 23:16 | | | NOVRVAL[1:0] | | | | | NON_OVERLAP |
| | | 31:24 | | | | | | | | |
| 0x08 | CLASSD_INTPMR | 7:0 | | ATTL[6:0] | | | | | | |
| | | 15:8 | | ATTR[6:0] | | | | | | |
| | | 23:16 | | FRAME[2:0] | | | SWAP | DEEMP | | DSPCLKFREQ |
| | | 31:24 | | MONOMODE[1:0] | | MONO | EQCFG[3:0] | | | |
| 0x0C | CLASSD_INTSR | 7:0 | | | | | | | | CFGERR |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | CLASSD_THR | 7:0 | LDATA[7:0] | | | | | | | |
| | | 15:8 | LDATA[15:8] | | | | | | | |
| | | 23:16 | RDATA[7:0] | | | | | | | |
| | | 31:24 | RDATA[15:8] | | | | | | | |
| 0x14 | CLASSD_IER | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | CLASSD_IDR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | CLASSD_IMR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | CLASSD_ISR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 ... 0xE3 | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xE4 | CLASSD_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |

### 42.7.1 CLASSD Control Register

**Name:** CLASSD_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SWRST |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets CLASSD, simulating a hardware reset. |

### 42.7.2 CLASSD Mode Register

**Name:** CLASSD_MR
**Offset:** 0x04
**Reset:** 0x00010022
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the CLASSD Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | NOVRVAL[1:0] | | | | | NON_OVERLAP |
| Access | | | R/W | R/W | | | | R/W |
| Reset | | | 0 | 0 | | | | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | PWMTYP |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | RMUTE | REN | | | LMUTE | LEN |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 1 | 0 | | | 1 | 0 |

**Bits 21:20 – NOVRVAL[1:0]** Non-Overlapping Value
This field has no effect when NON_OVERLAP = 0.

| Value | Name | Description |
|---|---|---|
| 0 | 5NS | Non-overlapping time is 5 ns |
| 1 | 10NS | Non-overlapping time is 10 ns |
| 2 | 15NS | Non-overlapping time is 15 ns |
| 3 | 20NS | Non-overlapping time is 20 ns |

**Bit 16 – NON_OVERLAP** Non-Overlapping Enable

| Value | Description |
|---|---|
| 0 | Non-overlapping circuit is disabled. |
| 1 | Non-overlapping circuit is enabled. |

**Bit 8 – PWMTYP** PWM Modulation Type
0 (TRAILING_EDGE): The signal is single-ended.

If NON_OVERLAP is cleared, the signal is sent to CLASSD_L0 and CLASSD_R0 (see figure Use Case 4A or figure Use Case 4B).

If NON_OVERLAP is set, the signal is sent to CLASSD_L0/L1 and CLASSD_R0/R1 (see figure Use Case 2).

1 (UNIFORM): The signal is differential.

If NON_OVERLAP is cleared, the signal is sent to CLASSD_L0/L2 and CLASSD_R0/R2 (see figure Use Case 3A or figure Use Case 3B).

If NON_OVERLAP is set, the signal is sent to CLASSD_L0/L1/L2/L3 and CLASSD_R0/R1/R2/R3 (see figure Use Case 1).

**Bit 5 – RMUTE** Right Channel Mute

| Value | Description |
|---|---|
| 0 | Right channel is unmuted. |
| 1 | Right channel is muted. |

**Bit 4 – REN** Right Channel Enable

| Value | Description |
|---|---|
| 0 | Right channel is disabled. |
| 1 | Right channel is enabled. |

**Bit 1 – LMUTE** Left Channel Mute

| Value | Description |
|---|---|
| 0 | Left channel is unmuted. |
| 1 | Left channel is muted. |

**Bit 0 – LEN** Left Channel Enable

| Value | Description |
|---|---|
| 0 | Left channel is disabled. |
| 1 | Left channel is enabled. |

### 42.7.3 CLASSD Interpolator Mode Register

**Name:** CLASSD_INTPMR
**Offset:** 0x08
**Reset:** 0x00304E4E
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the CLASSD Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | MONOMODE[1:0] | | MONO | EQCFG[3:0] | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | FRAME[2:0] | | | SWAP | DEEMP | | DSPCLKFREQ |
| Access | | R/W | R/W | R/W | R/W | R/W | | R/W |
| Reset | | 0 | 1 | 1 | 0 | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | ATTR[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 1 | 0 | 0 | 1 | 1 | 1 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | ATTL[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 1 | 0 | 0 | 1 | 1 | 1 | 0 |

**Bits 30:29 – MONOMODE[1:0]** Mono Mode Selection
This field defines which signal is sent to both channels when the MONO bit is set.

| Value | Name | Description |
|---|---|---|
| 0 | MONOMIX | (left + right) / 2 is sent to both channels |
| 1 | MONOSAT | (left + right) is sent to both channels. If the sum is too high, the result is saturated. |
| 2 | MONOLEFT | THR[15:0] is sent to both the left and the right channels |
| 3 | MONORIGHT | THR[31:16] is sent to both the left and the right channels |

**Bit 28 – MONO** Mono Signal
0 (DISABLED): The signal is sent stereo to the left and right channels.

1 (ENABLED): The same signal is sent to both the left and the right channels. The sent signal is defined by the MONOMODE field value.

**Bits 27:24 – EQCFG[3:0]** Equalization Selection
EQCFG field values 13–15 = flat response

| Value | Name | Description |
|---|---|---|
| 0 | FLAT | Flat response |
| 1 | BBOOST12 | Bass boost +12 dB |
| 2 | BBOOST6 | Bass boost +6 dB |
| 3 | BCUT12 | Bass cut -12 dB |
| 4 | BCUT6 | Bass cut -6 dB |
| 5 | MBOOST3 | Medium boost +3 dB |
| 6 | MBOOST8 | Medium boost +8 dB |
| 7 | MCUT3 | Medium cut -3 dB |
| 8 | MCUT8 | Medium cut -8 dB |
| 9 | TBOOST12 | Treble boost +12 dB |
| 10 | TBOOST6 | Treble boost +6 dB |
| 11 | TCUT12 | Treble cut -12 dB |
| 12 | TCUT6 | Treble cut -6 dB |

**Bits 22:20 – FRAME[2:0]** CLASSD Incoming Data Sampling Frequency

| Value | Name | Description |
|---|---|---|
| 0 | FRAME_8K | 8 kHz |
| 1 | FRAME_16K | 16 kHz |
| 2 | FRAME_32K | 32 kHz |
| 3 | FRAME_48K | 48 kHz |
| 4 | FRAME_96K | 96 kHz |
| 5 | FRAME_22K | 22.05 kHz |
| 6 | FRAME_44K | 44.1 kHz |
| 7 | FRAME_88K | 88.2 kHz |

**Bit 19 – SWAP** Swap Left and Right Channels
0 (LEFT_ON_LSB): Left channel is on CLASSD_THR[15:0], right channel is on CLASSD_THR[31:16].

1 (RIGHT_ON_LSB): Right channel is on CLASSD_THR[15:0], left channel is on CLASSD_THR[31:16].

**Bit 18 – DEEMP** Enable De-emphasis Filter
0 (DISABLED): De-emphasis filter is disabled.

1 (ENABLED): De-emphasis filter is enabled.

**Bit 16 – DSPCLKFREQ** DSP Clock Frequency
0 (12M288): DSP Clock (DSPCLK) is 12.288 MHz.

1 (11M2896): DSP Clock (DSPCLK) is 11.2896 MHz.

**Bits 14:8 – ATTR[6:0]** Right Channel Attenuation
Right channel attenuation is defined as follows:

– if ATTR ≤ 77 the attenuation is -ATTR dB

– else the right signal is muted

**Bits 6:0 – ATTL[6:0]** Left Channel Attenuation
Left channel attenuation is defined as follows:

– if ATTL ≤ 77 the attenuation is -ATTL dB

– else the left signal is muted

### 42.7.4 CLASSD Interpolator Status Register

**Name:** CLASSD_INTSR
**Offset:** 0x0C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CFGERR |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 0 – CFGERR** Configuration Error

| Value | Description |
|---|---|
| 0 | The frame and clock configurations are correct. |
| 1 | The frame and clock configurations are wrong (see Clock Configuration for information about allowed configurations). |

### 42.7.5 CLASSD Transmit Holding Register

**Name:** CLASSD_THR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RDATA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RDATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LDATA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LDATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – RDATA[15:0]** Right Channel Data

**Bits 15:0 – LDATA[15:0]** Left Channel Data

### 42.7.6 CLASSD Interrupt Enable Register

**Name:** CLASSD_IER
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – DATRDY** Data Ready

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the interrupt when CLASSD is ready to receive new data to convert. |

### 42.7.7 CLASSD Interrupt Disable Register

**Name:** CLASSD_IDR
**Offset:** 0x18
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – DATRDY** Data Ready

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the interrupt when CLASSD is ready to receive new data to convert. |

### 42.7.8 CLASSD Interrupt Mask Register

**Name:** CLASSD_IMR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 0 – DATRDY** Data Ready

| Value | Description |
|---|---|
| 0 | The interrupt is disabled. |
| 1 | The interrupt is enabled. |

### 42.7.9 CLASSD Interrupt Status Register

**Name:** CLASSD_ISR
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 0 – DATRDY** Data Ready

| Value | Description |
|---|---|
| 0 | CLASSD has not been ready to convert a value since the last read of CLASSD_ISR. |
| 1 | CLASSD has been ready to convert a value since the last read of CLASSD_ISR. |

### 42.7.10 CLASSD Write Protection Mode Register

**Name:** CLASSD_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x434C44 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit.<br>Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See Register Write Protection for the list of registers that can be write-protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x434C44 (“CLD” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x434C44 (“CLD” in ASCII). |

# 43. Inter-IC Sound Controller (I2SC)

## 43.1 Description

The Inter-IC Sound Controller (I2SC) provides a 5-wire, bidirectional, synchronous, digital audio link to external audio devices: I2SC_DI, I2SC_DO, I2SC_WS, I2SC_CK, and I2SC_MCK pins.

The I2SC is compliant with the Inter-IC Sound ($I^2S$) bus specification.

The I2SC consists of a receiver, a transmitter and a common clock generator that can be enabled separately to provide Master, Slave or Controller modes with receiver and/or transmitter active.

DMA Controller channels, separate for the receiver and for the transmitter, allow a continuous high bit rate data transfer without processor intervention to the following:

- Audio CODECs in Master, Slave, or Controller mode
- Stereo DAC or ADC through a dedicated $I^2S$ serial interface

The I2SC uses a single DMA Controller channel for both audio channels.

The 8- and 16-bit compact stereo format reduces the required DMA Controller bandwidth by transferring the left and right samples within the same data word.

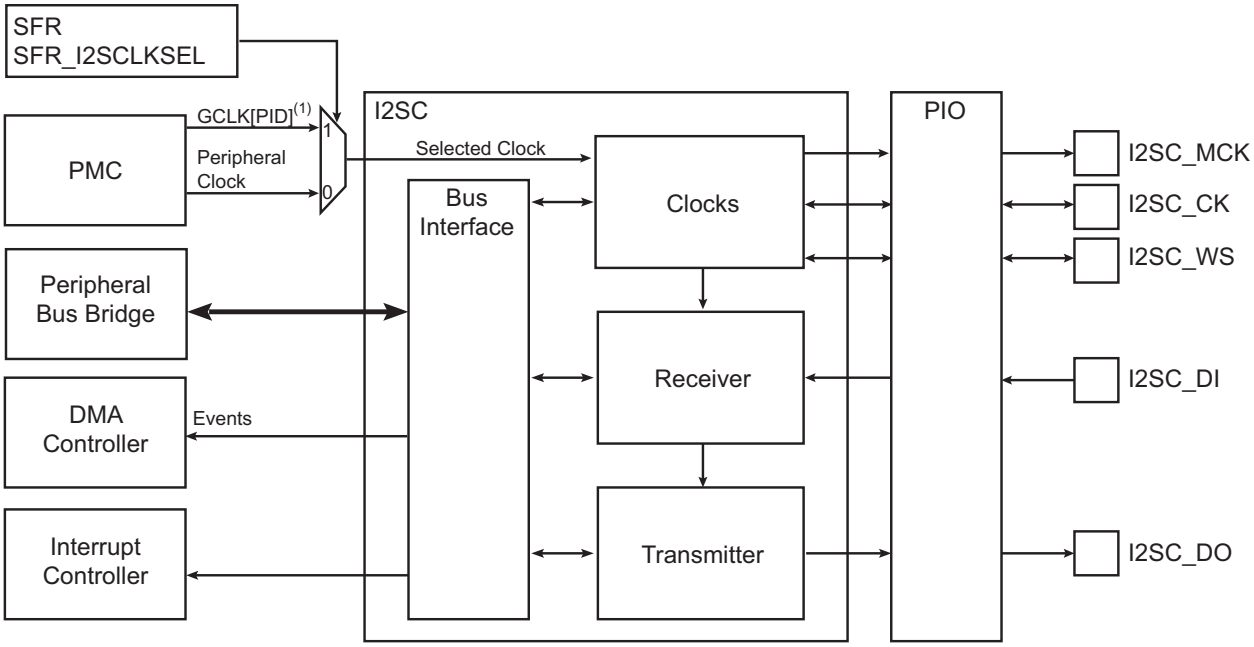
In Master mode, the I2SC can produce a 32 $f_s$ to 1024 $f_s$ master clock that provides an over-sampling clock to an external audio codec or digital signal processor (DSP).

## 43.2 Embedded Characteristics

- Compliant with Inter-IC Sound ($I^2S$) Bus Specification
- Master, Slave, and Controller Modes
  - Slave: Data Received/Transmitted
  - Master: Data Received/Transmitted And Clocks Generated
  - Controller: Clocks Generated
- Individual Enable and Disable of Receiver, Transmitter and Clocks
- Configurable Clock Generator Common to Receiver and Transmitter
  - Suitable for a Wide Range of Sample Frequencies ($f_s$), Including 32 kHz, 44.1 kHz, 48 kHz, 88.2 kHz, 96 kHz, and 192 kHz
  - 32 $f_s$ to 1024 $f_s$ Master Clock Generated for External Oversampling Data Converters
- Support for Multiple Data Formats
  - 32-, 24-, 20-, 18-, 16-, and 8-bit Mono or Stereo Format
  - 16- and 8-bit Compact Stereo Format, with Left and Right Samples Packed in the Same Word to Reduce Data Transfers
- DMA Controller Interfaces the Receiver and Transmitter to Reduce Processor Overhead
  - One DMA Controller Channel for Both Audio Channels
- Smart Holding Registers Management to Avoid Audio Channels Mix After Overrun or Underrun

## 43.3 Block Diagram

**Figure 43-1. I2SC Block Diagram**




(1) For the value of 'PID', refer to I2SCx in the table "Peripheral Identifiers".

## 43.4 I/O Lines Description

**Table 43-1. I/O Lines Description**

| Pin Name | Pin Description | Type |
|---|---|---|
| I2SC_MCK | Master Clock | Output |
| I2SC_CK | Serial Clock | Input/Output |
| I2SC_WS | $I^2S$ Word Select | Input/Output |
| I2SC_DI | Serial Data Input | Input |
| I2SC_DO | Serial Data Output | Output |

## 43.5 Product Dependencies

To use the I2SC, other parts of the system must be configured correctly, as described below.

### 43.5.1 I/O Lines

The I2SC pins may be multiplexed with I/O Controller lines. The user must first program the PIO Controller to assign the required I2SC pins to their peripheral function. If the I2SC I/O lines are not used by the application, they can be used for other purposes by the PIO Controller. The user must enable the I2SC inputs and outputs that are used.

### 43.5.2 Power Management

If the CPU enters a Sleep mode that disables clocks used by the I2SC, the I2SC stops functioning and resumes operation after the system wakes up from Sleep mode.

### 43.5.3 Clocks

The clock for the I2SC bus interface is generated by the Power Management Controller (PMC). I2SC must be disabled before disabling the clock to avoid freezing the I2SC in an undefined state.

### 43.5.4 DMA Controller

The I2SC interfaces to the DMA Controller. Using the I2SC DMA functionality requires the DMA Controller to be programmed first.

### 43.5.5 Interrupt Sources

The I2SC interrupt line is connected to the Interrupt Controller. Using the I2SC interrupt requires the Interrupt Controller to be programmed first.

## 43.6 Functional Description

### 43.6.1 Initialization

The I2SC features a receiver, a transmitter and a clock generator for Master and Controller modes. Receiver and transmitter share the same serial clock and word select.

Before enabling the I2SC, the selected configuration must be written to the I2SC Mode Register (I2SC_MR) and to the I2S Clock Source Selection register (SFR_I2SCLKSEL) described in the section “Special Function Registers (SFR)”.

If the I2SC_MR.IMCKMODE bit is set, the I2SC_MR.IMCKFS field must be configured as described in section “Serial Clock and Word Select Generation”.

Once the I2SC_MR has been written, the I2SC clock generator, receiver, and transmitter can be enabled by writing a ’1’ to the CKEN, RXEN, and TXEN bits in the Control Register (I2SC_CR). The clock generator can be enabled alone in Controller mode to output clocks to the I2SC_MCK, I2SC_CK, and I2SC_WS pins. The clock generator must also be enabled if the receiver or the transmitter is enabled.

The clock generator, receiver, and transmitter can be disabled independently by writing a ’1’ to I2SC_CR.CXDIS, I2SC_CR.RXDIS and/or I2SC_CR.TXDIS, respectively. Once requested to stop, they stop only when the transmission of the pending frame transmission is completed.

### 43.6.2 Basic Operation

The receiver can be operated by reading the Receiver Holding Register (I2SC_RHR), whenever the Receive Ready (RXRDY) bit in the Status Register (I2SC_SR) is set. Successive values read from RHR correspond to the samples from the left and right audio channels for the successive frames.

The transmitter can be operated by writing to the Transmitter Holding Register (I2SC_THR), whenever the Transmit Ready (TXRDY) bit in the I2SC_SR is set. Successive values written to THR correspond to the samples from the left and right audio channels for the successive frames.

The RXRDY and TXRDY bits can be polled by reading the I2SC_SR.

The I2SC processor load can be reduced by enabling interrupt-driven operation. The RXRDY and/or TXRDY interrupt requests can be enabled by writing a ’1’ to the corresponding bit in the Interrupt Enable Register (I2SC_IER). The interrupt service routine associated to the I2SC interrupt request is executed whenever the Receive Ready or the Transmit Ready status bit is set.

### 43.6.3 Master, Controller and Slave Modes

In Master and Controller modes, the I2SC provides the master clock, the serial clock and the word select. I2SC_MCK, I2SC_CK, and I2SC_WS pins are outputs.

In Controller mode, the I2SC receiver and transmitter are disabled. Only the clocks are enabled and used by an external receiver and/or transmitter.

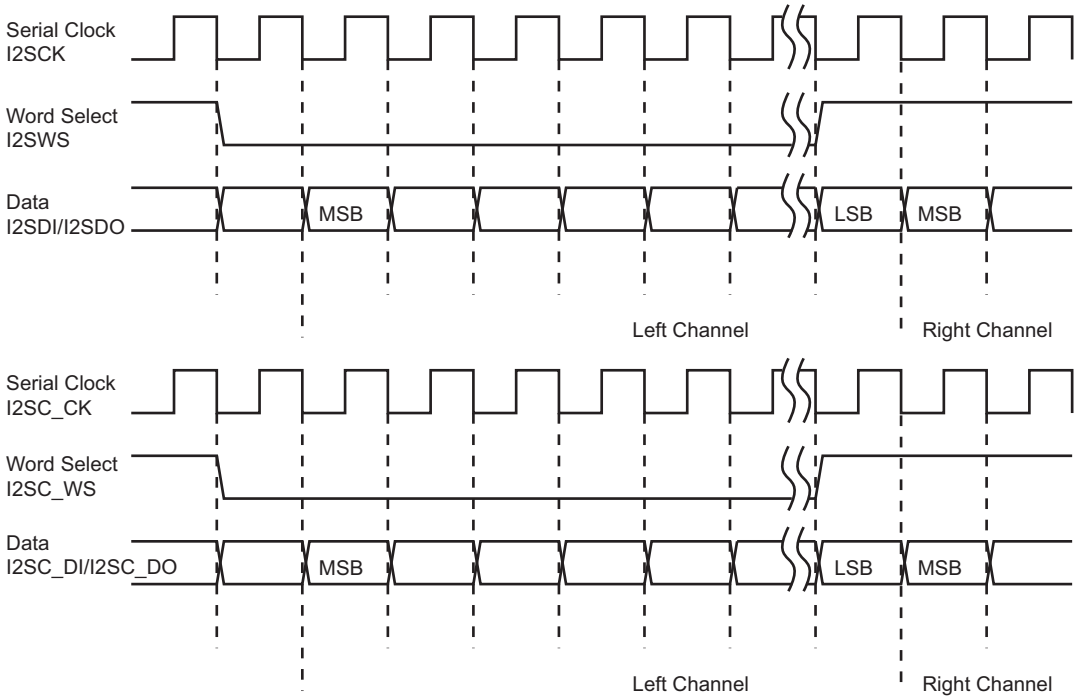
In Slave mode, the I2SC receives the serial clock and the word select from an external master. I2SC_CK and I2SC_WS pins are inputs.

The mode is selected by writing the MODE field in the I2SC_MR. Since the MODE field changes the direction of the I2SC_WS and I2SC_SCK pins, the I2SC_MR must be written when the I2SC is stopped.

### 43.6.4 I²S Reception and Transmission Sequence

As specified in the I²S protocol, data bits are left-justified in the word select time slot, with the MSB transmitted first, starting one clock period after the transition on the word select line.

**Figure 43-2. I²S Reception and Transmission Sequence**




Data bits are sent on the falling edge of the serial clock and sampled on the rising edge of the serial clock. The word select line indicates the channel in transmission, a low level for the left channel and a high level for the right channel.

The length of transmitted words can be chosen among 8, 16, 18, 20, 24, and 32 bits by writing the I2SC_MR.DATALENGTH field.

If the time slot allows for more data bits than written in the I2SC_MR.DATALENGTH field, zeroes are appended to the transmitted data word or extra received bits are discarded.

### 43.6.5 Serial Clock and Word Select Generation

The generation of clocks in the I2SC is described in figure ”I2SC Clock Generation”.

In Slave mode, the serial clock and word select clock are driven by an external master. I2SC_CK and I2SC_WS pins are inputs.

In Master mode, the user can configure the master clock, serial clock, and word select clock through the I2SC_MR. I2SC_MCK, I2SC_CK, and I2SC_WS pins are outputs and MCK is used to derive the I2SC clocks.

In Master mode, if the peripheral clock frequency is higher than 96 MHz, GCLK[PID] from the PMC must be selected as the I2SC input clock by writing a ’1’ in the CLKSELx bit of the SFR_I2SCLKSEL register located in SFR.

Audio codecs connected to the I2SC pins may require a master clock (I2SC_MCK) signal with a frequency multiple of the audio sample frequency ($f_s$), such as $256f_s$. When the I2SC is in Master mode, writing a ’1’ to I2SC_MR.IMCKMODE outputs MCK as master clock to the I2SC_MCK pin, and divides MCK to create the internal bit clock, output on the I2SC_CK pin. The clock division factor is defined by writing to I2SC_MR.IMCKFS and I2SC_MR.DATALENGTH, as described in the I2SC_MR.IMCKFS field description.

The master clock (I2SC_MCK) frequency is (2×16 × (IMCKFS + 1)) / (IMCKDIV + 1) times the sample frequency ($f_s$), i.e., I2SC_WS frequency.

Example: If the sampling rate is 44.1 kHz with an I2S master clock (I2SC_MCK) ratio of 256, the core frequency must be an integer multiple of 11.2896 MHz. Assuming an integer multiple of 4, the IMCKDIV field must be configured to 4; the field IMCKFS must then be set to 31.

The serial clock (I2SC_CK) frequency is 2 × Slot Length times the sample frequency ($f_s$), where Slot Length is defined in table Slot Length.

**Table 43-2. Slot Length**

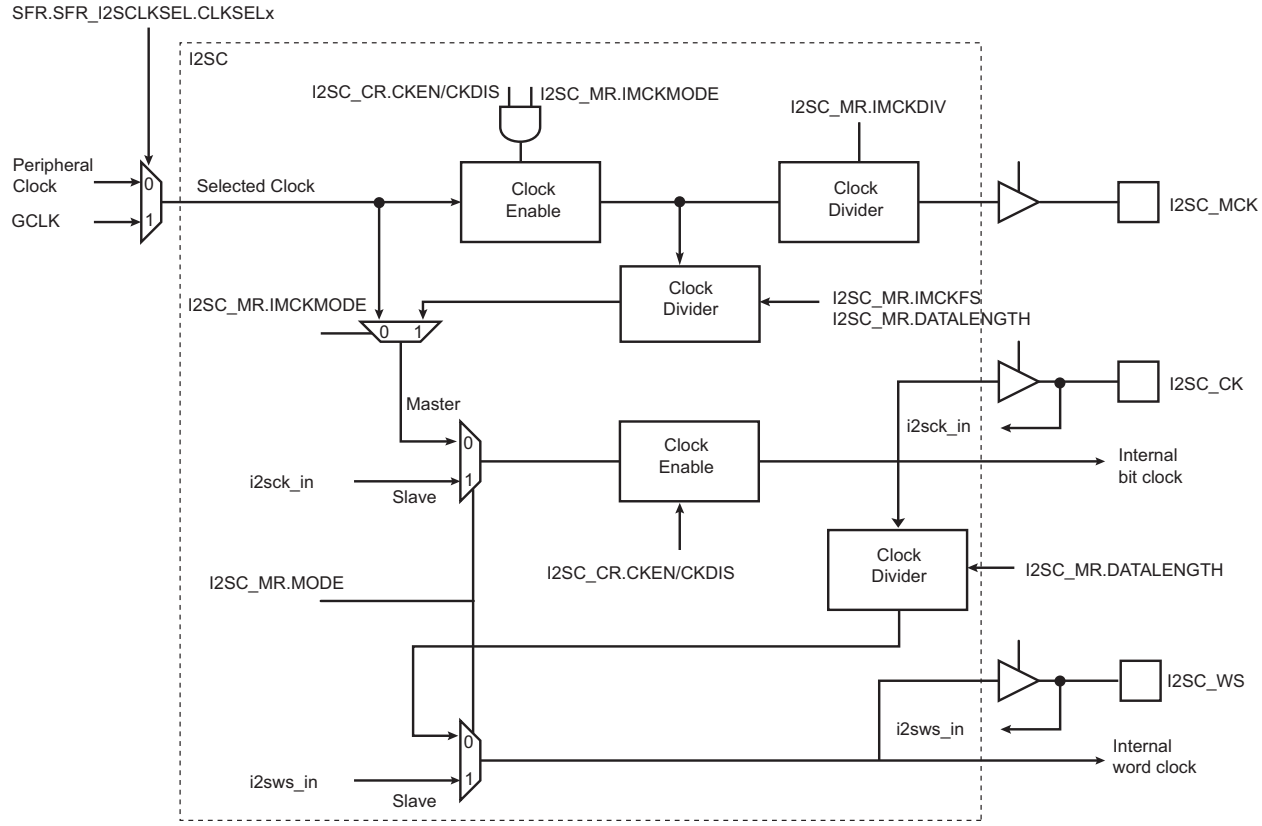
| I2SC_MR.DATALENGTH | Word Length | Slot Length |
|---|---|---|
| 0 | 32 bits | 32 |
| 1 | 24 bits | 32 if I2SC_MR.IWS = 0<br>24 if I2SC_MR.IWS = 1 |
| 2 | 20 bits | |
| 3 | 18 bits | |
| 4 | 16 bits | 16 |
| 5 | 16 bits compact stereo | |
| 6 | 8 bits | 8 |
| 7 | 8 bits compact stereo | |

**⚠WARNING** I2SC_MR.IMCKMODE must be written to ’1’ if the master clock frequency is strictly higher than the serial clock.

If a master clock output is not required, the MCK clock is used as I2SC_CK by clearing I2SC_MR.IMCKMODE. Alternatively, if the frequency of the MCK clock used is a multiple of the required I2SC_CK frequency, the I2SC_MCK to I2SC_CK divider can be used with the ratio defined by writing the I2SC_MR.IMCKFS field.

The I2SC_WS pin is used as word select as described in section “I2S Reception and Transmission Sequence”.

**Figure 43-3. I2SC Clock Generation**




### 43.6.6 Mono

When the Transmit Mono bit (TXMONO) in I2SC_MR is set, data written to the left channel is duplicated to the right output channel.

When the Receive Mono bit (RXMONO) in I2SC_MR is set, data received from the left channel is duplicated to the right channel.

### 43.6.7 Holding Registers

The I2SC user interface includes a Receive Holding Register (I2SC_RHR) and a Transmit Holding Register (I2SC_THR). These registers are used to access audio samples for both audio channels.

When a new data word is available in I2SC_RHR, the Receive Ready bit (RXRDY) in I2SC_SR is set. Reading I2SC_RHR clears this bit.

A receive overrun condition occurs if a new data word becomes available before the previous data word has been read from I2SC_RHR. In this case, the Receive Overrun bit in I2SC_SR and bit i of the RXORCH field in I2SC_SR are set, where i is the current receive channel number.

When I2SC_THR is empty, the Transmit Ready bit (TXRDY) in I2SC_SR is set. Writing to I2SC_THR clears this bit.

A transmit underrun condition occurs if a new data word needs to be transmitted before it has been written to I2SC_THR. In this case, the Transmit Underrun (TXUR) bit and bit i of the TXORCH field in I2SC_SR are set, where i is the current transmit channel number. If the TXSAME bit in I2SC_MR is ’0’, then a zero data word is transmitted in case of underrun. If I2SC_MR.TXSAME is ’1’, then the previous data word for the current transmit channel number is transmitted.

Data words are right-justified in I2SC_RHR and I2SC_THR. For the 16-bit compact stereo data format, the left sample uses bits 15:0 and the right sample uses bits 31:16 of the same data word. For the 8-bit compact stereo data format, the left sample uses bits 7:0 and the right sample uses bits 15:8 of the same data word.

### 43.6.8 DMA Controller Operation

All receiver audio channels are assigned to a single DMA Controller.

The DMA Controller reads from the I2SC_RHR and writes to the I2SC_THR for both audio channels successively.

The DMA Controller transfers may use 32-bit word, 16-bit halfword, or 8-bit byte depending on the value of the I2SC_MR.DATALENGTH field.

### 43.6.9 Loopback Mode

For debug purposes, the I2SC can be configured to loop back the transmitter to the Receiver. Writing a ’1’ to the I2SC_MR.LOOP bit internally connects I2SC_DO to I2SC_DI, so that the transmitted data is also received. Writing a ’0’ to I2SC_MR.LOOP restores the normal behavior with independent Receiver and Transmitter. As for other changes to the Receiver or Transmitter configuration, the I2SC Receiver and Transmitter must be disabled before writing to I2SC_MR to update I2SC_MR.LOOP.

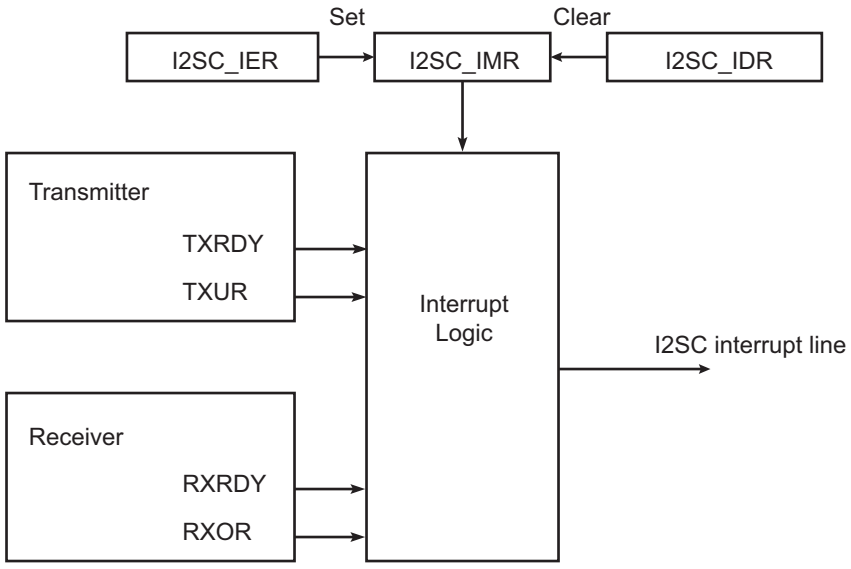
### 43.6.10 Interrupts

An I2SC interrupt request can be triggered whenever one or several of the following bits are set in I2SC_SR: Receive Ready (RXRDY), Receive Overrun (RXOR), Transmit Ready (TXRDY) or Transmit Underrun (TXUR).

The interrupt request is generated if the corresponding bit in the Interrupt Mask Register (I2SC_IMR) is set. Bits in I2SC_IMR are set by writing a ’1’ to the corresponding bit in I2SC_IER and cleared by writing a ’1’ to the corresponding bit in the Interrupt Disable Register (I2SC_IDR). The interrupt request remains active until the corresponding bit in I2SC_SR is cleared by writing a ’1’ to the corresponding bit in the Status Clear Register (I2SC_SCR).

For debug purposes, interrupt requests can be simulated by writing a ’1’ to the corresponding bit in the Status Set Register (I2SC_SSR).

**Figure 43-4. Interrupt Block Diagram**



## 43.7 I2SC Application Examples

The I2SC supports several serial communication modes used in audio or high-speed serial links. Examples of standard applications are shown in the following figures. All serial link applications supported by the I2SC are not listed here.

**Figure 43-5. Slave Transmitter I2SC Application Example**

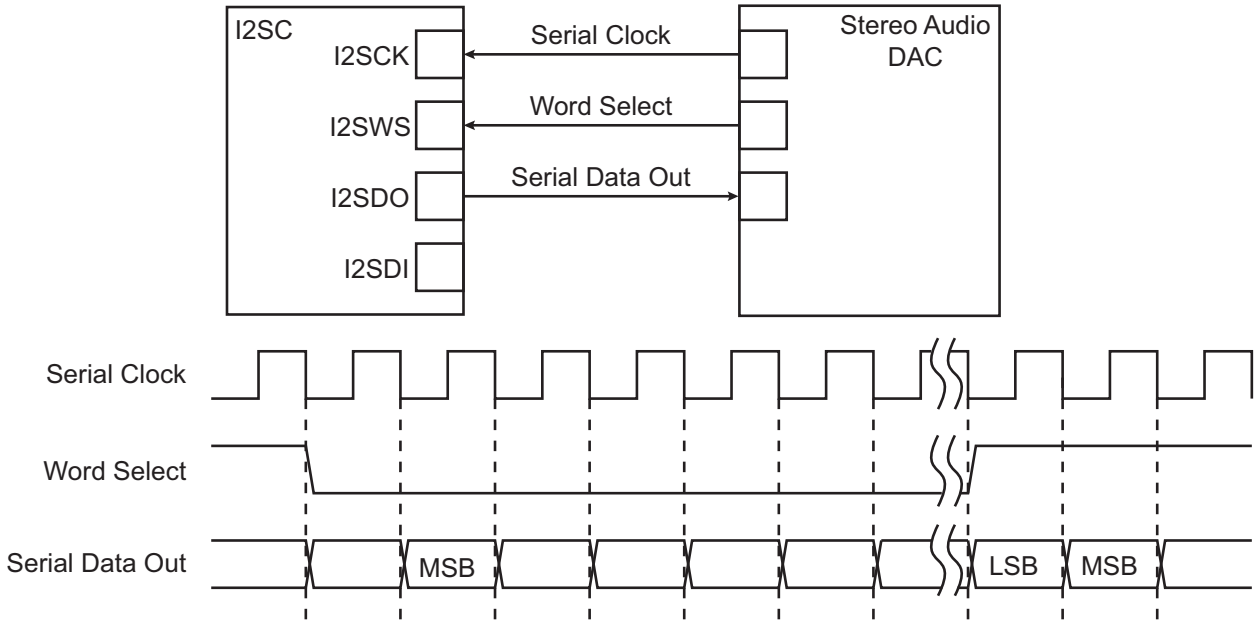
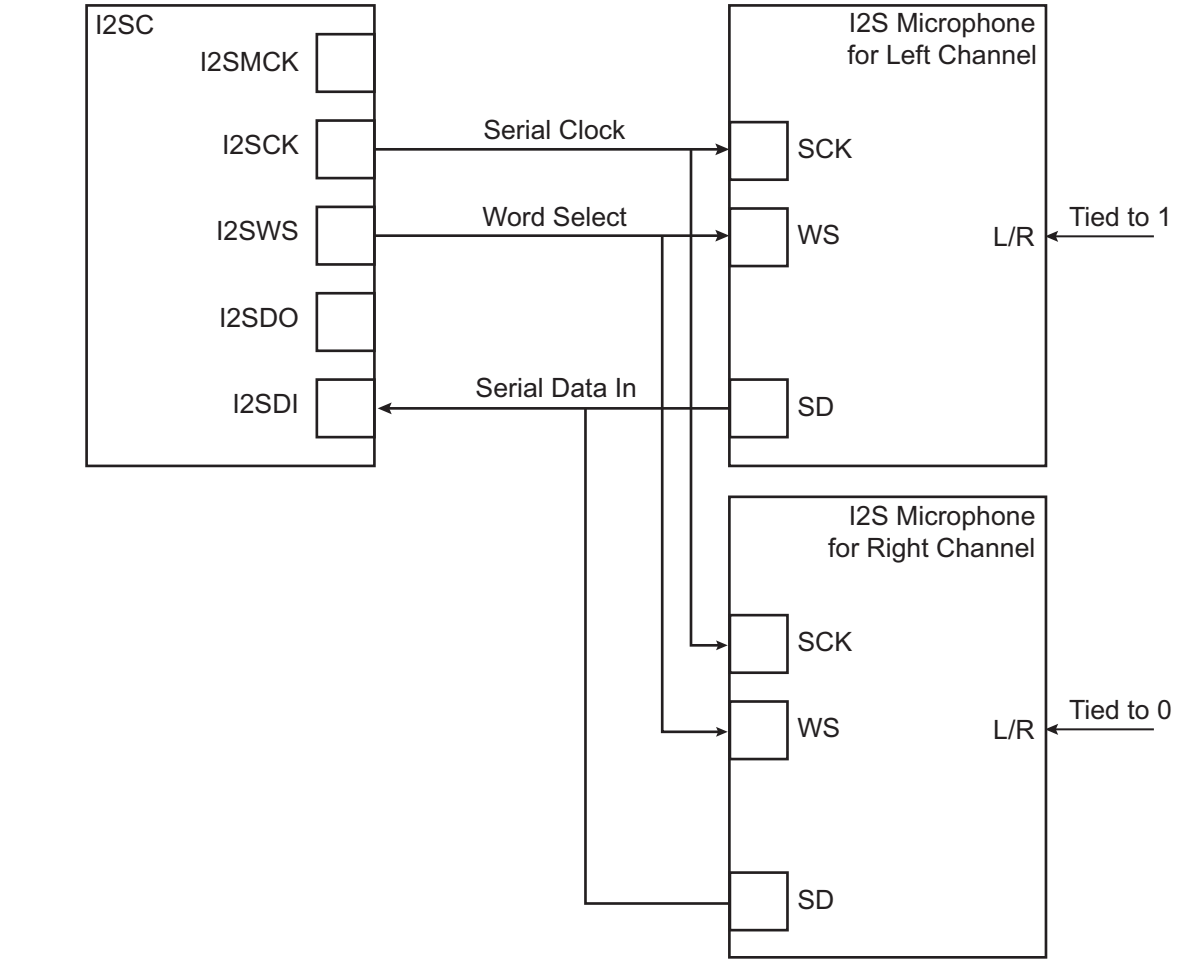


**Figure 43-6. Dual Microphone Application Block Diagram**



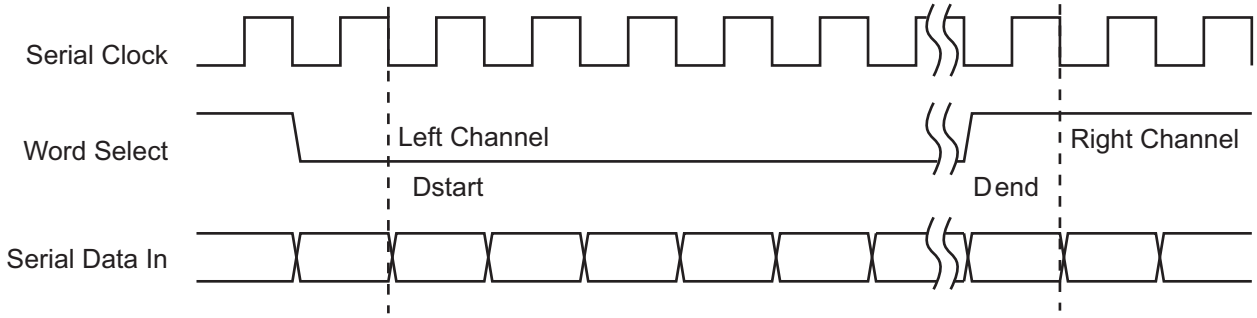
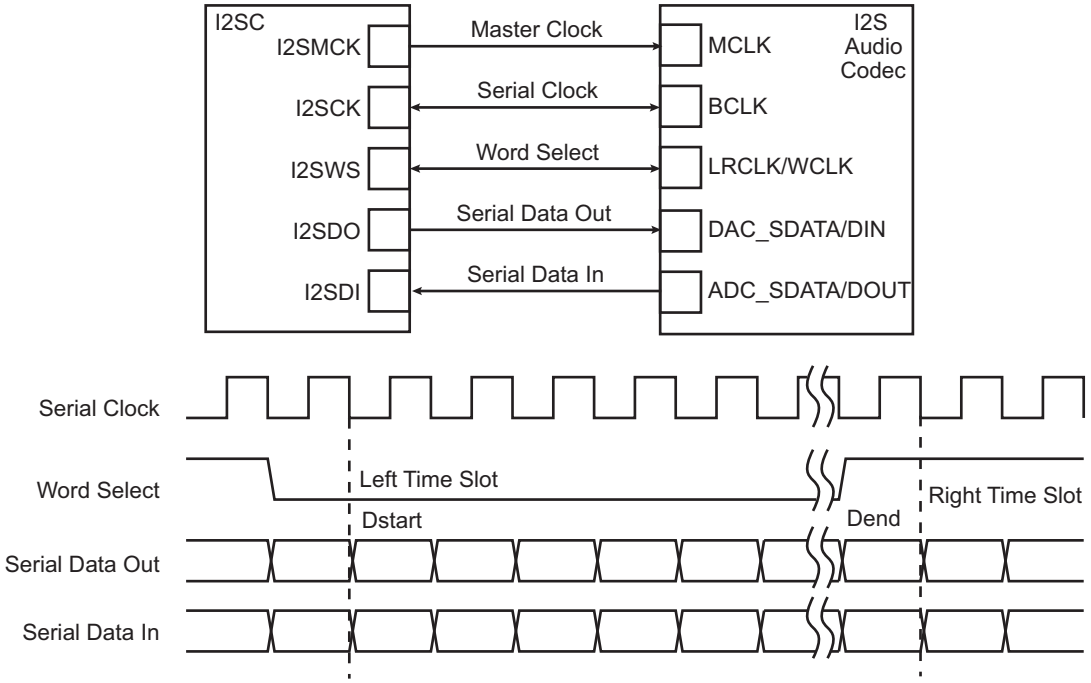



**Figure 43-7. Codec Application Block Diagram**



## 43.8 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | I2SC_CR | 7:0 | SWRST | | TXDIS | TXEN | CKDIS | CKEN | RXDIS | RXEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | I2SC_MR | 7:0 | FORMAT[1:0] | | | DATALENGTH[2:0] | | | | MODE |
| | | 15:8 | | TXSAME | | TXMONO | | RXLOOP | | RXMONO |
| | | 23:16 | | | IMCKDIV[5:0] | | | | | |
| | | 31:24 | IWS | IMCKMODE | IMCKFS[5:0] | | | | | |
| 0x08 | I2SC_SR | 7:0 | | TXUR | TXRDY | TXEN | | RXOR | RXRDY | RXEN |
| | | 15:8 | | | | | | | RXORCH[1:0] | |
| | | 23:16 | | | TXURCH[1:0] | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | I2SC_SCR | 7:0 | | TXUR | | | | RXOR | | |
| | | 15:8 | | | | | | | RXORCH[1:0] | |
| | | 23:16 | | | TXURCH[1:0] | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | I2SC_SSR | 7:0 | | TXUR | | | | RXOR | | |
| | | 15:8 | | | | | | | RXORCH[1:0] | |
| | | 23:16 | | | TXURCH[1:0] | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | I2SC_IER | 7:0 | | TXUR | TXRDY | | | RXOR | RXRDY | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | I2SC_IDR | 7:0 | | TXUR | TXRDY | | | RXOR | RXRDY | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | I2SC_IMR | 7:0 | | TXUR | TXRDY | | | RXOR | RXRDY | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | I2SC_RHR | 7:0 | RHR[7:0] | | | | | | | |
| | | 15:8 | RHR[15:8] | | | | | | | |
| | | 23:16 | RHR[23:16] | | | | | | | |
| | | 31:24 | RHR[31:24] | | | | | | | |
| 0x24 | I2SC_THR | 7:0 | THR[7:0] | | | | | | | |
| | | 15:8 | THR[15:8] | | | | | | | |
| | | 23:16 | THR[23:16] | | | | | | | |
| | | 31:24 | THR[31:24] | | | | | | | |

### 43.8.1 I2SC Control Register

**Name:** I2SC_CR
**Offset:** 0x00
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWRST | | TXDIS | TXEN | CKDIS | CKEN | RXDIS | RXEN |
| Access | W | | W | W | W | W | W | W |
| Reset | – | | – | – | – | – | – | – |

**Bit 7 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit resets all the registers in the I2SC. The I2SC is disabled after the reset. |

**Bit 5 – TXDIS** Transmitter Disable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit disables the I2SC transmitter. Bit I2SC_SR.TXEN is cleared when the Transmitter is stopped. |

**Bit 4 – TXEN** Transmitter Enable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit enables the I2SC transmitter, if TXDIS is not one. Bit I2SC_SR.TXEN is set when the Transmitter is started. |

**Bit 3 – CKDIS** Clocks Disable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a zone to this bit disables the I2SC clock generation. |

**Bit 2 – CKEN** Clocks Enable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit enables the I2SC clocks generation, if CKDIS is not one. |

**Bit 1 – RXDIS** Receiver Disable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit disables the I2SC receiver. Bit I2SC_SR.RXEN is cleared when the receiver is stopped. |

**Bit 0 – RXEN** Receiver Enable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit enables the I2SC receiver, if RXDIS is not one. Bit I2SC_SR.RXEN is set when the receiver is activated. |

### 43.8.2 I2SC Mode Register

**Name:** I2SC_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

The I2SC_MR must be written when the I2SC is stopped. The proper sequence is to write to I2SC_MR, then write to I2SC_CR to enable the I2SC or to disable the I2SC before writing a new value to I2SC_MR.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IWS | IMCKMODE | IMCKFS[5:0] | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | IMCKDIV[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | TXSAME | | TXMONO | | RXLOOP | | RXMONO |
| Access | | R/W | | R/W | | R/W | | R/W |
| Reset | | 0 | | 0 | | 0 | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FORMAT[1:0] | | | DATALENGTH[2:0] | | | | MODE |
| Access | R/W | R/W | | R/W | R/W | R/W | | R/W |
| Reset | 0 | 0 | | 0 | 0 | 0 | | 0 |

**Bit 31 – IWS** I2SC_WS Slot Width
Refer to table Slot Length (I2S format).

| Value | Description |
|---|---|
| 0 | I2SC_WS slot is 32 bits wide for DATALENGTH = 18/20/24 bits. |
| 1 | I2SC_WS slot is 24 bits wide for DATALENGTH = 18/20/24 bits. |

**Bit 30 – IMCKMODE** Master Clock Mode

**WARNING** If I2SC_MCK frequency is the same as I2SC_CK, IMCKMODE must be cleared. Refer to section Serial Clock and Word Select Generation and table Slot Length.

| Value | Description |
|---|---|
| 0 | No master clock generated (Selected Clock drives I2SC_CK output). |
| 1 | Master clock generated (internally generated clock is used as I2SC_MCK output). |

**Bits 29:24 – IMCKFS[5:0]** Master Clock to $f_s$ Ratio
Master clock frequency is [2 x 16 × (IMCKFS + 1)] / (IMCKDIV + 1) times the sample rate, i.e., I2SC_WS frequency.

| Value | Name | Description |
|---|---|---|
| 0 | M2SF32 | Sample frequency ratio set to 32 |
| 1 | M2SF64 | Sample frequency ratio set to 64 |
| 2 | M2SF96 | Sample frequency ratio set to 96 |
| 3 | M2SF128 | Sample frequency ratio set to 128 |
| 5 | M2SF192 | Sample frequency ratio set to 192 |
| 7 | M2SF256 | Sample frequency ratio set to 256 |
| 11 | M2SF384 | Sample frequency ratio set to 384 |
| 15 | M2SF512 | Sample frequency ratio set to 512 |
| 23 | M2SF768 | Sample frequency ratio set to 768 |
| 31 | M2SF1024 | Sample frequency ratio set to 1024 |
| 47 | M2SF1536 | Sample frequency ratio set to 1536 |
| 63 | M2SF2048 | Sample frequency ratio set to 2048 |

**Bits 21:16 – IMCKDIV[5:0]** Selected Clock to I2SC Master Clock Ratio
I2SC_MCK Master clock output frequency is Selected Clock divided by (IMCKDIV + 1). Refer to the IMCKFS field description.

**Note:**

1. This field is write-only. Always read as ‘0’.
2. Do not write a ‘0’ to this field.

**Bit 14 – TXSAME** Transmit Data when Underrun

| Value | Description |
|---|---|
| 0 | Zero sample transmitted when underrun. |
| 1 | Previous sample transmitted when underrun |

**Bit 12 – TXMONO** Transmit Mono

| Value | Description |
|---|---|
| 0 | Stereo |
| 1 | Mono, with left audio samples duplicated to right audio channel by the I2SC. |

**Bit 10 – RXLOOP** Loopback Test Mode

| Value | Description |
|---|---|
| 0 | Normal mode |
| 1 | I2SC_DO output of I2SC is internally connected to I2SC_DI input. |

**Bit 8 – RXMONO** Receive Mono

| Value | Description |
|---|---|
| 0 | Stereo |
| 1 | Mono, with left audio samples duplicated to right audio channel by the I2SC. |

**Bits 7:6 – FORMAT[1:0]** Data Format

| Value | Name | Description |
|---|---|---|
| 0 | I2S | $I^2S$ format, stereo with I2SC_WS low for left channel, and MSB of sample starting one I2SC_CK period after I2SC_WS edge |
| 1 | LJ | Left-justified format, stereo with I2SC_WS high for left channel, and MSB of sample starting on I2SC_WS edge |
| 2 | – | Reserved |
| 3 | – | Reserved |

**Bits 4:2 – DATALENGTH[2:0]** Data Word Length

| Value | Name | Description |
|---|---|---|
| 0 | 32_BITS | Data length is set to 32 bits. |
| 1 | 24_BITS | Data length is set to 24 bits. |
| 2 | 20_BITS | Data length is set to 20 bits. |
| 3 | 18_BITS | Data length is set to 18 bits. |
| 4 | 16_BITS | Data length is set to 16 bits. |
| 5 | 16_BITS_COMPACT | Data length is set to 16-bit compact stereo. Left sample in bits 15:0 and right sample in bits 31:16 of same word. |
| 6 | 8_BITS | Data length is set to 8 bits. |
| 7 | 8_BITS_COMPACT | Data length is set to 8-bit compact stereo. Left sample in bits 7:0 and right sample in bits 15:8 of the same word. |

**Bit 0 – MODE** Inter-IC Sound Controller Mode

| Value | Name | Description |
|---|---|---|
| 0 | SLAVE | I2SC_CK and I2SC_WS pin inputs used as bit clock and word select/frame synchronization. |
| 1 | MASTER | Bit clock and word select/frame synchronization generated by I2SC from MCK and output to I2SC_CK and I2SC_WS pins. Peripheral clock or GCLK is output as master clock on I2SC_MCK if I2SC_MR.IMCKMODE is set. |

### 43.8.3 I2SC Status Register

**Name:** I2SC_SR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | TXURCH[1:0] | | | | | |
| Access | | | R | R | | | | |
| Reset | | | 0 | 0 | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXORCH[1:0] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TXUR | TXRDY | TXEN | | RXOR | RXRDY | RXEN |
| Access | | R | R | R | | R | R | R |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bits 21:20 – TXURCH[1:0]** Transmit Underrun Channel

| Value | Description |
|---|---|
| 0 | This field is cleared when I2SC_SCR.TXUR is written to ’1’. |
| 1 | Bit i of this field is set when a transmit underrun error occurred in channel i (i = 0 for first channel of the frame). |

**Bits 9:8 – RXORCH[1:0]** Receive Overrun Channel
This field is cleared when I2SC_SCR.RXOR is written to ’1’.

Bit i of this field is set when a receive overrun error occurred in channel i (i = 0 for first channel of the frame).

**Bit 6 – TXUR** Transmit Underrun

| Value | Description |
|---|---|
| 0 | This bit is cleared when the corresponding bit in I2SC_SCR is written to ’1’. |
| 1 | This bit is set when an underrun error occurs on I2SC_THR or when the corresponding bit in I2SC_SSR is written to ’1’. |

**Bit 5 – TXRDY** Transmit Ready

| Value | Description |
|---|---|
| 0 | This bit is cleared when data is written to I2SC_THR. |
| 1 | This bit is set when I2SC_THR is empty and can be written with new data to be transmitted. |

**Bit 4 – TXEN** Transmitter Enabled

| Value | Description |
|---|---|
| 0 | This bit is cleared when the transmitter is disabled, following a I2SC_CR.TXDIS or I2SC_CR.SWRST request. |
| 1 | This bit is set when the transmitter is enabled, following a I2SC_CR.TXEN request. |

**Bit 2 – RXOR** Receive Overrun

| Value | Description |
|---|---|
| 0 | This bit is cleared when the corresponding bit in I2SC_SCR is written to ’1’. |
| 1 | This bit is set when an overrun error occurs on I2SC_RHR or when the corresponding bit in I2SC_SSR is written to ’1’. |

**Bit 1 – RXRDY** Receive Ready

| Value | Description |
|---|---|
| 0 | This bit is cleared when I2SC_RHR is read. |
| 1 | This bit is set when received data is present in I2SC_RHR. |

**Bit 0 – RXEN** Receiver Enabled

| Value | Description |
|---|---|
| 0 | This bit is cleared when the receiver is disabled, following a RXDIS or SWRST request in I2SC_CR. |
| 1 | This bit is set when the receiver is enabled, following a RXEN request in I2SC_CR. |

### 43.8.4 I2SC Status Clear Register

**Name:** I2SC_SCR
**Offset:** 0x0C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | TXURCH[1:0] | | | | | |
| Access | | | W | W | | | | |
| Reset | | | – | – | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXORCH[1:0] | |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TXUR | | | | RXOR | | |
| Access | | W | | | | W | | |
| Reset | | – | | | | – | | |

**Bits 21:20 – TXURCH[1:0]** Transmit Underrun Per Channel Status Clear
Writing a ’0’ has no effect.

Writing a ’1’ to any bit in this field clears the corresponding bit in the I2SC_SR and the corresponding interrupt request.

**Bits 9:8 – RXORCH[1:0]** Receive Overrun Per Channel Status Clear
Writing a ’0’ has no effect.

Writing a ’1’ to any bit in this field clears the corresponding bit in the I2SC_SR and the corresponding interrupt request.

**Bit 6 – TXUR** Transmit Underrun Status Clear
Writing a ’0’ to this bit has no effect.

Writing a ’1’ to this bit clears the status bit.

**Bit 2 – RXOR** Receive Overrun Status Clear
Writing a ’0’ to this bit has no effect.

Writing a ’1’ to this bit clears the status bit.

### 43.8.5 I2SC Status Set Register

**Name:** I2SC_SSR
**Offset:** 0x10
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | TXURCH[1:0] | | | | | |
| Access | | | W | W | | | | |
| Reset | | | – | – | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXORCH[1:0] | |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TXUR | | | | RXOR | | |
| Access | | W | | | | W | | |
| Reset | | – | | | | – | | |

**Bits 21:20 – TXURCH[1:0]** Transmit Underrun Per Channel Status Set
Writing a ’0’ has no effect.

Writing a ’1’ to any bit in this field sets the corresponding bit in I2SC_SR and the corresponding interrupt request.

**Bits 9:8 – RXORCH[1:0]** Receive Overrun Per Channel Status Set
Writing a ’0’ has no effect.

Writing a ’1’ to any bit in this field sets the corresponding bit in I2SC_SR and the corresponding interrupt request.

**Bit 6 – TXUR** Transmit Underrun Status Set
Writing a ’0’ to this bit has no effect.

Writing a ’1’ to this bit sets the status bit.

**Bit 2 – RXOR** Receive Overrun Status Set
Writing a ’0’ to this bit has no effect.

Writing a ’1’ to this bit sets the status bit.

### 43.8.6 I2SC Interrupt Enable Register

**Name:** I2SC_IER
**Offset:** 0x14
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TXUR | TXRDY | | | RXOR | RXRDY | |
| Access | | W | W | | | W | W | |
| Reset | | – | – | | | – | – | |

**Bit 6 – TXUR** Transmit Underflow Interrupt Enable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit sets the corresponding bit in I2SC_IMR. |

**Bit 5 – TXRDY** Transmit Ready Interrupt Enable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit as no effect. |
| 1 | Writing a ’1’ to this bit sets the corresponding bit in I2SC_IMR. |

**Bit 2 – RXOR** Receiver Overrun Interrupt Enable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit sets the corresponding bit in I2SC_IMR. |

**Bit 1 – RXRDY** Receiver Ready Interrupt Enable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit sets the corresponding bit in I2SC_IMR. |

### 43.8.7 I2SC Interrupt Disable Register

**Name:** I2SC_IDR
**Offset:** 0x18
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TXUR | TXRDY | | | RXOR | RXRDY | |
| Access | | W | W | | | W | W | |
| Reset | | – | – | | | – | – | |

**Bit 6 – TXUR** Transmit Underflow Interrupt Disable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit clears the corresponding bit in I2SC_IMR. |

**Bit 5 – TXRDY** Transmit Ready Interrupt Disable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit clears the corresponding bit in I2SC_IMR. |

**Bit 2 – RXOR** Receiver Overrun Interrupt Disable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit clears the corresponding bit in I2SC_IMR. |

**Bit 1 – RXRDY** Receiver Ready Interrupt Disable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit clears the corresponding bit in I2SC_IMR. |

### 43.8.8 I2SC Interrupt Mask Register

**Name:** I2SC_IMR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TXUR | TXRDY | | | RXOR | RXRDY | |
| Access | | R | R | | | R | R | |
| Reset | | 0 | 0 | | | 0 | 0 | |

**Bit 6 – TXUR** Transmit Underflow Interrupt Disable

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is disabled. This bit is cleared when the corresponding bit in I2SC_IDR is written to ’1’. |
| 1 | The corresponding interrupt is enabled. This bit is set when the corresponding bit in I2SC_IER is written to ’1’. |

**Bit 5 – TXRDY** Transmit Ready Interrupt Disable

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is disabled. This bit is cleared when the corresponding bit in I2SC_IDR is written to ’1’. |
| 1 | The corresponding interrupt is enabled. This bit is set when the corresponding bit in I2SC_IER is written to ’1’. |

**Bit 2 – RXOR** Receiver Overrun Interrupt Disable

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is disabled. This bit is cleared when the corresponding bit in I2SC_IDR is written to ’1’. |

| Value | Description |
|---|---|
| 1 | The corresponding interrupt is enabled. This bit is set when the corresponding bit in I2SC_IER is written to ’1’. |

**Bit 1 – RXRDY** Receiver Ready Interrupt Disable

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is disabled. This bit is cleared when the corresponding bit in I2SC_IDR is written to ’1’. |
| 1 | The corresponding interrupt is enabled. This bit is set when the corresponding bit in I2SC_IER is written to ’1’. |

### 43.8.9 I2SC Receiver Holding Register

**Name:** I2SC_RHR
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RHR[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RHR[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RHR[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RHR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RHR[31:0]** Receiver Holding Register
This field is set by hardware to the last received data word. If I2SC_MR.DATALENGTH specifies fewer than 32 bits, data is right-justified in the RHR field.

### 43.8.10 I2SC Transmitter Holding Register

**Name:** I2SC_THR
**Offset:** 0x24
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | THR[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | THR[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | THR[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | THR[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 31:0 – THR[31:0]** Transmitter Holding Register
Next data word to be transmitted after the current word if TXRDY is not set. If I2SC_MR.DATALENGTH specifies fewer than 32 bits, data is right-justified in the THR field.

# 44. Synchronous Serial Controller (SSC)

## 44.1 Description

The Synchronous Serial Controller (SSC) provides a synchronous communication link with external devices. It supports many serial synchronous communication protocols generally used in audio and telecom applications such as I2S, Short Frame Sync, Long Frame Sync, etc.

The SSC contains an independent receiver and transmitter and a common clock divider. The receiver and the transmitter each interface with three signals: the TD/RD signal for data, the TK/RK signal for the clock and the TF/RF signal for the Frame Sync. The transfers can be programmed to start automatically or on different events detected on the Frame Sync signal.

The SSC high-level of programmability and its use of DMA enable a continuous high bit rate data transfer without processor intervention.

Featuring connection to the DMA, the SSC enables interfacing with low processor overhead to:

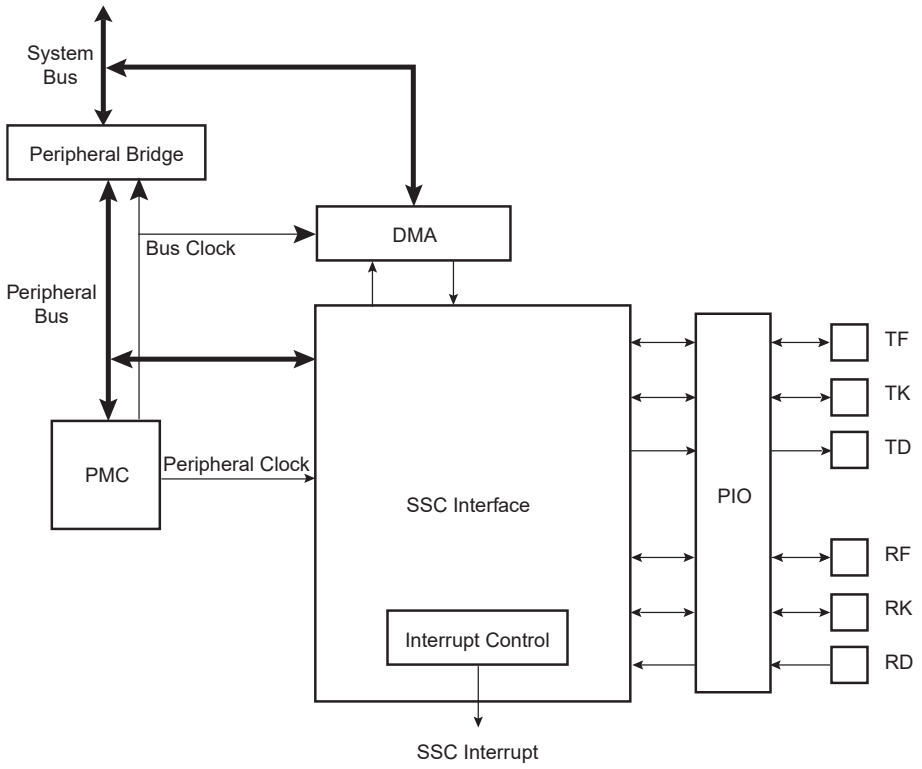
- Codecs in Master or Slave mode
- DAC through dedicated serial interface, particularly I2S
- Magnetic card reader

## 44.2 Embedded Characteristics

- Provides Serial Synchronous Communication Links Used in Audio and Telecom Applications
- Contains an Independent Receiver and Transmitter and a Common Clock Divider
- Interfaced with the DMA Controller (DMAC) to Reduce Processor Overhead
- Offers a Configurable Frame Sync and Data Length
- Receiver and Transmitter Can be Programmed to Start Automatically or on Detection of Different Events on the Frame Sync Signal
- Receiver and Transmitter Include a Data Signal, a Clock Signal and a Frame Sync Signal

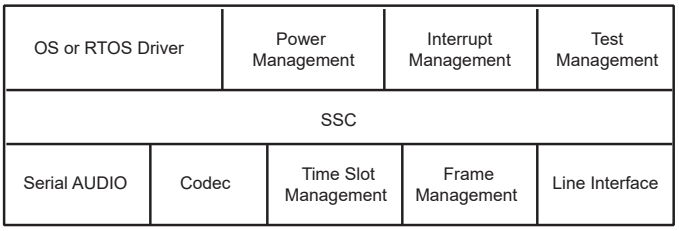
## 44.3 Block Diagram

**Figure 44-1. Block Diagram**




## 44.4 Application Block Diagram

**Figure 44-2. Application Block Diagram**




## 44.5 SSC Application Examples

The SSC can support several serial communication modes used in audio or high speed serial links. Some standard applications are shown in the following figures. All serial link applications supported by the SSC are not listed here.

**Figure 44-3. Audio Application Block Diagram**

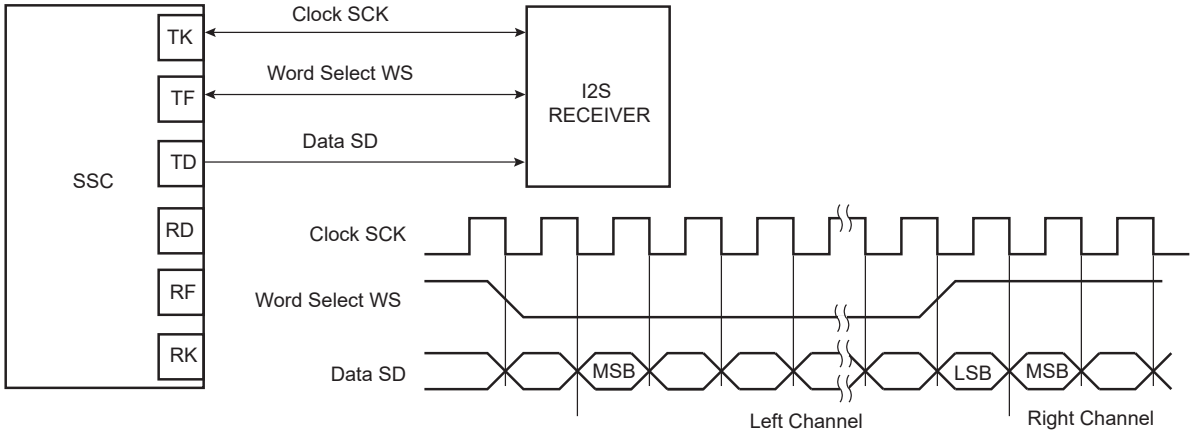



**Figure 44-4. Codec Application Block Diagram**

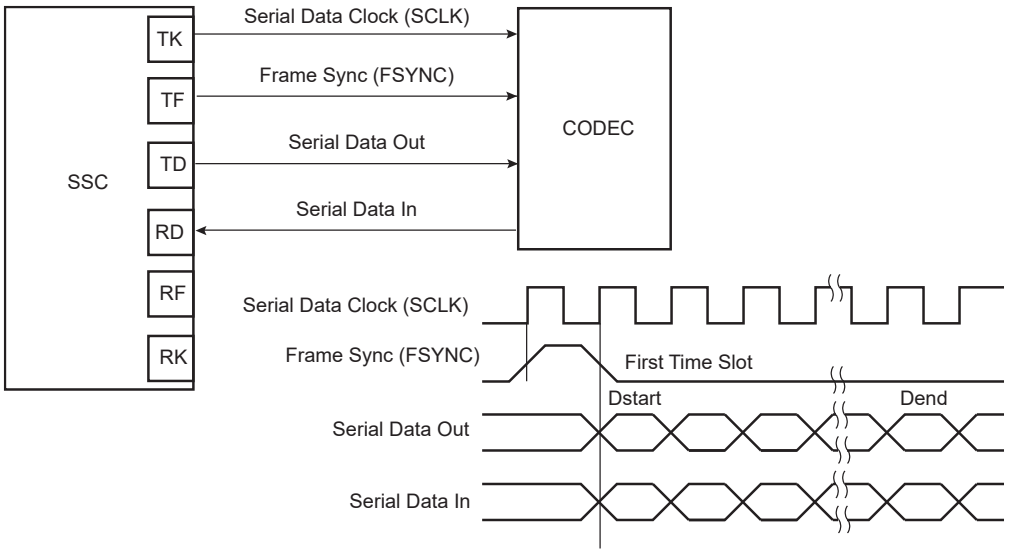
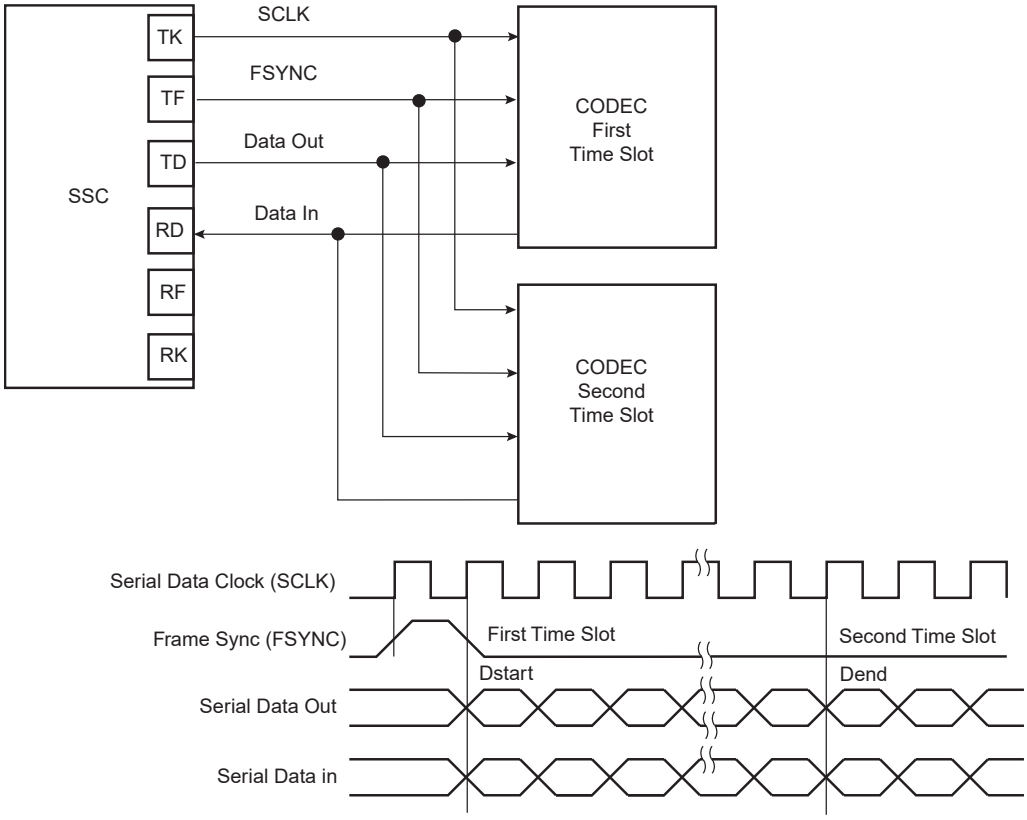


**Figure 44-5. Time Slot Application Block Diagram**




## 44.6 Pin Name List

**Table 44-1. I/O Lines Description**

| Pin Name | Pin Description | Type |
|---|---|---|
| RF | Receive Frame Synchronization | Input/Output |
| RK | Receive Clock | Input/Output |
| RD | Receive Data | Input |
| TF | Transmit Frame Synchronization | Input/Output |
| TK | Transmit Clock | Input/Output |
| TD | Transmit Data | Output |

## 44.7 Product Dependencies

### 44.7.1 I/O Lines

The pins used for interfacing the compliant external devices may be multiplexed with PIO lines.

Before using the SSC receiver, the PIO controller must be configured to dedicate the SSC receiver I/O lines to the SSC Peripheral mode.

Before using the SSC transmitter, the PIO controller must be configured to dedicate the SSC transmitter I/O lines to the SSC Peripheral mode.

### 44.7.2 Power Management

The SSC is not continuously clocked. The SSC interface may be clocked through the Power Management Controller (PMC), therefore the programmer must first configure the PMC to enable the SSC clock.

### 44.7.3 Interrupt

The SSC interface has an interrupt line connected to the interrupt controller. Handling interrupts requires programming the interrupt controller before configuring the SSC.

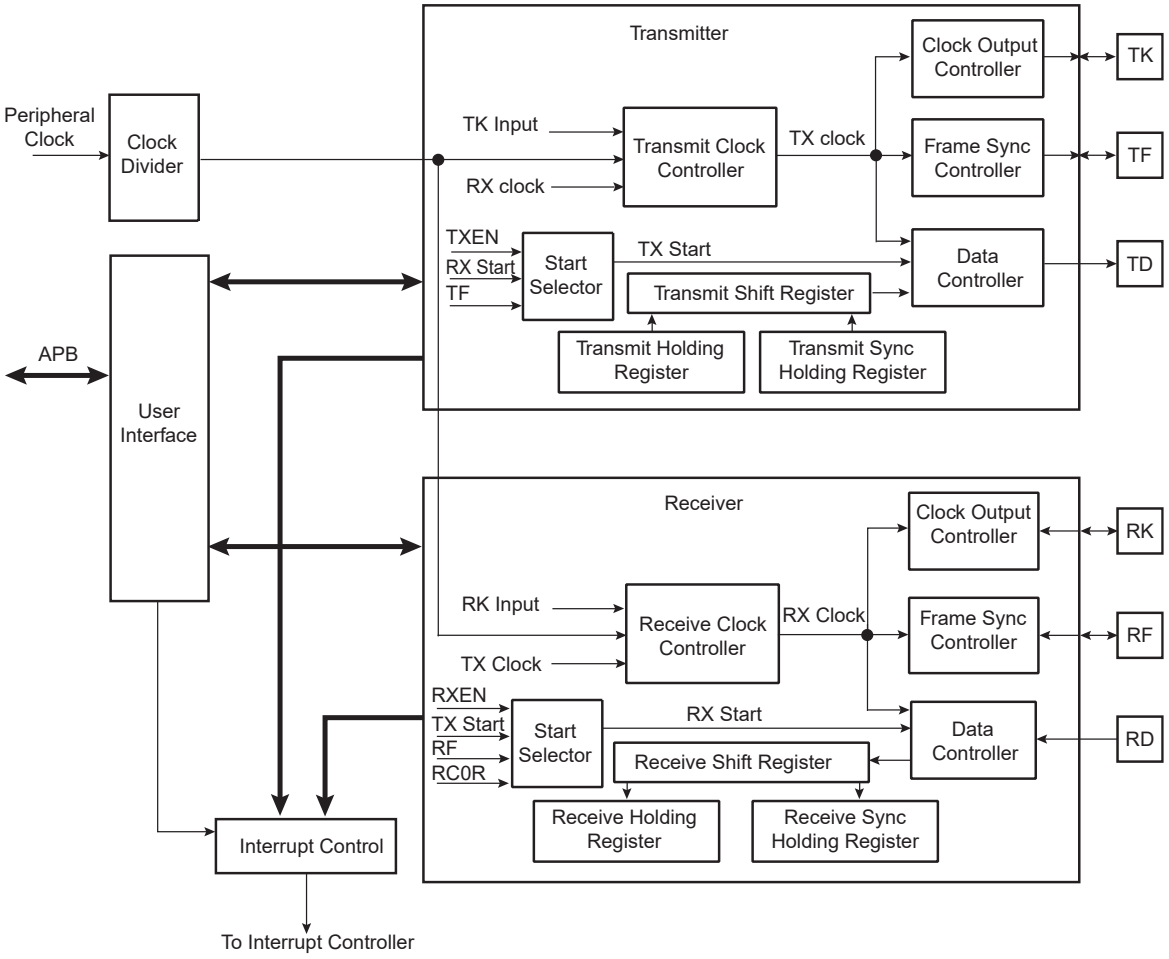
All SSC interrupts can be enabled/disabled configuring the SSC Interrupt Mask Register. Each pending and unmasked SSC interrupt asserts the SSC interrupt line. The SSC interrupt service routine can get the interrupt origin by reading the SSC Interrupt Status Register.

## 44.8 Functional Description

This section contains the functional description of the following: SSC Functional Block, Clock Management, Data Format, Start, Transmit, Receive and Frame Synchronization.

The receiver and transmitter operate separately. However, they can work synchronously by programming the receiver to use the transmit clock and/or to start a data transfer when transmission starts. Alternatively, this can be done by programming the transmitter to use the receive clock and/or to start a data transfer when reception starts. The transmitter and the receiver can be programmed to operate with the clock signals provided on either the TK or RK pins. This allows the SSC to support many Slave mode data transfers. The maximum clock speed allowed on the TK and RK pins is the peripheral clock divided by 2.

**Figure 44-6. SSC Functional Block Diagram**




### 44.8.1 Clock Management

The transmit clock can be generated by:

- an external clock received on the TK I/O pad
- the receive clock
- the internal clock divider

The receive clock can be generated by:

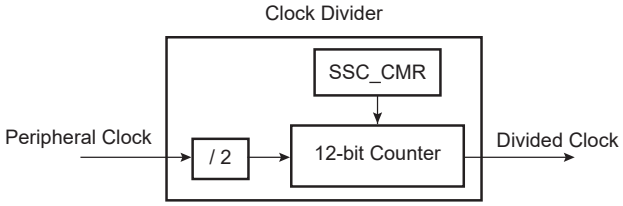
- an external clock received on the RK I/O pad
- the transmit clock
- the internal clock divider

Furthermore, the transmitter block can generate an external clock on the TK I/O pad, and the receive block can generate an external clock on the RK I/O pad.

This allows the SSC to support many Master and Slave mode data transfers.

#### 44.8.1.1 Clock Divider

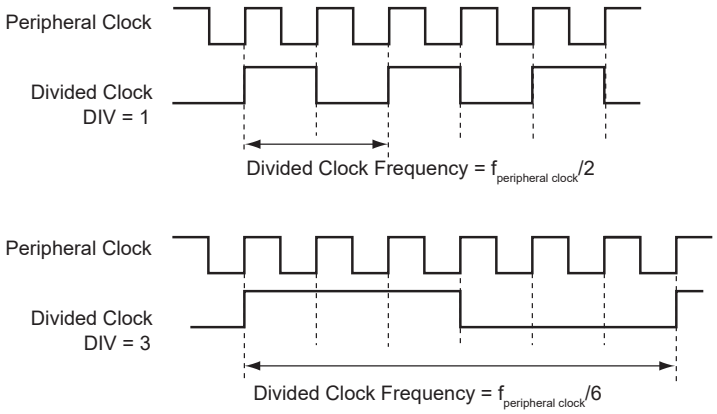
**Figure 44-7. Divided Clock Block Diagram**




The peripheral clock divider is determined by the 12-bit field DIV counter and comparator (so its maximal value is 4095) in the Clock Mode Register (SSC_CMR), allowing a peripheral clock division by up to 8190. The Divided Clock is provided to both the receiver and the transmitter. When this field is programmed to 0, the Clock Divider is not used and remains inactive.

When DIV is set to a value equal to or greater than 1, the Divided Clock has a frequency of peripheral clock divided by 2 times DIV. Each level of the Divided Clock has a duration of the peripheral clock multiplied by DIV. This ensures a 50% duty cycle for the Divided Clock regardless of whether the DIV value is even or odd.

**Figure 44-8. Divided Clock Generation**

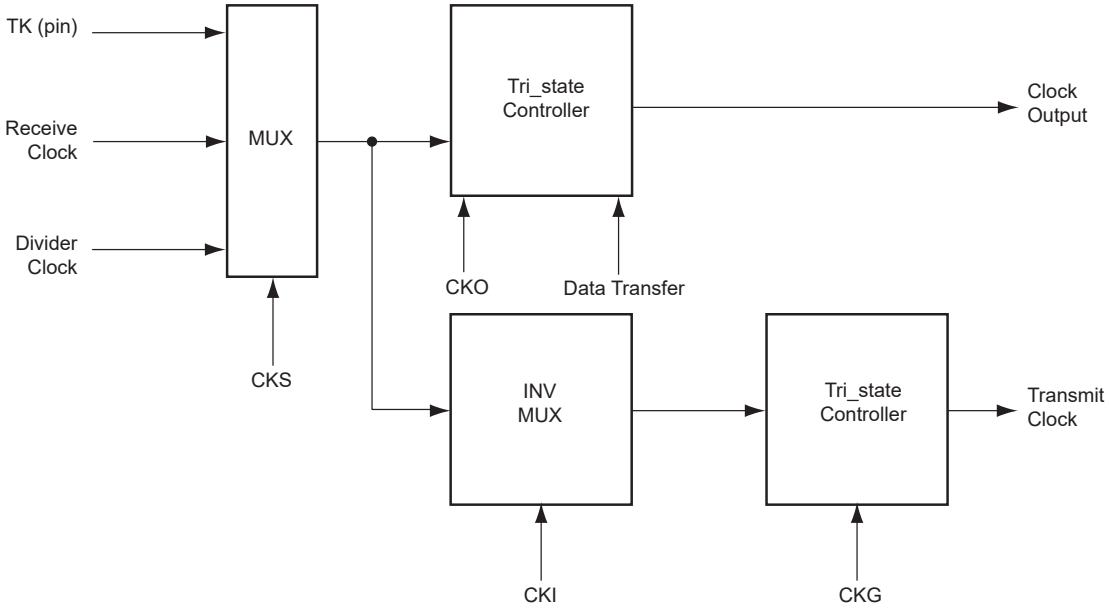



#### 44.8.1.2 Transmit Clock Management

The transmit clock is generated from the receive clock or the divider clock or an external clock scanned on the TK I/O pad. The transmit clock is selected by the CKS field in the Transmit Clock Mode Register (SSC_TCMR). Transmit Clock can be inverted independently by the CKI bits in the SSC_TCMR.

The transmitter can also drive the TK I/O pad continuously or be limited to the current data transfer. The clock output is configured by the SSC_TCMR. The Transmit Clock Inversion (CKI) bits have no effect on the clock outputs. Programming the SSC_TCMR to select TK pin (CKS field) and at the same time Continuous Transmit Clock (CKO field) can lead to unpredictable results.

**Figure 44-9. Transmit Clock Management**

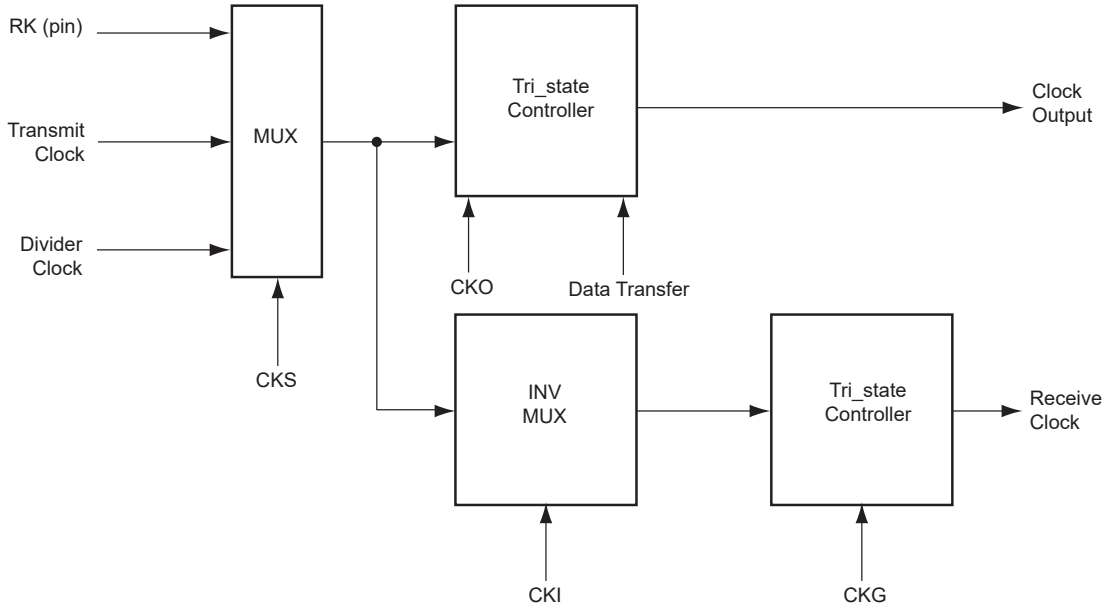



### 44.8.1.3 Receive Clock Management

The receive clock is generated from the transmit clock or the divider clock or an external clock scanned on the RK I/O pad. The Receive Clock is selected by the CKS field in SSC_RCMR (Receive Clock Mode Register). Receive Clocks can be inverted independently by the CKI bits in SSC_RCMR.

The receiver can also drive the RK I/O pad continuously or be limited to the current data transfer. The clock output is configured by the SSC_RCMR. The Receive Clock Inversion (CKI) bits have no effect on the clock outputs. Programming the SSC_RCMR to select RK pin (CKS field) and at the same time Continuous Receive Clock (CKO field) can lead to unpredictable results.

**Figure 44-10. Receive Clock Management**



#### 44.8.1.4 Serial Clock Ratio Considerations

The transmitter and the receiver can be programmed to operate with the clock signals provided on either the TK or RK pins. This allows the SSC to support many Slave mode data transfers. In this case, the maximum clock speed allowed on the RK pin is:

- Peripheral clock divided by 2 if Receive Frame Synchronization is input
- Peripheral clock divided by 3 if Receive Frame Synchronization is output

In addition, the maximum clock speed allowed on the TK pin is:

- Peripheral clock divided by 6 if Transmit Frame Synchronization is input
- Peripheral clock divided by 2 if Transmit Frame Synchronization is output

These are only theoretical speed limits for first order calculations. Exact speed limits on TK and RK are provided in the "Electrical Characteristics" chapter.

### 44.8.2 Transmit Operations

A transmit frame is triggered by a start event and can be followed by synchronization data before data transmission.

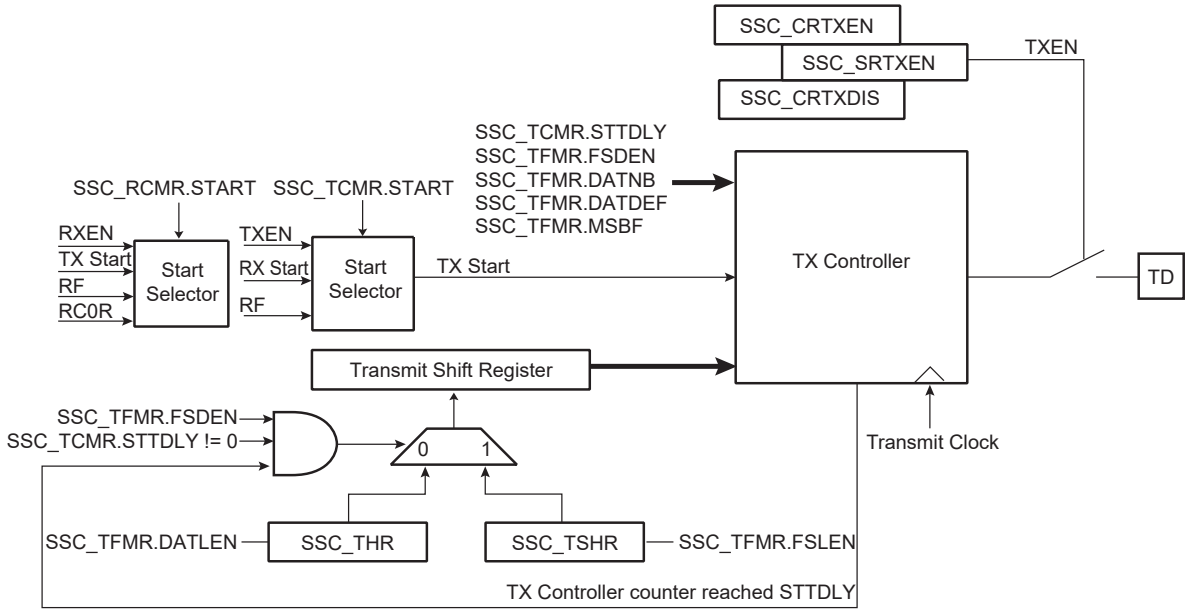
The start event is configured by setting the SSC_TCMR. See Start.

The frame synchronization is configured setting the Transmit Frame Mode Register (SSC_TFMR). See Frame Synchronization.

To transmit data, the transmitter uses a shift register clocked by the transmit clock signal and the start mode selected in the SSC_TCMR. Data is written by the application to the Transmit Holding register (SSC_THR) then transferred to the transmit shift register according to the data format selected.

When both the SSC_THR and the transmit shift register are empty, the status flag TXEMPTY is set in the Status register (SSC_SR). When the Transmit Holding register is transferred in the transmit shift register, the status flag TXRDY is set in the SSC_SR and additional data can be loaded in the Transmit Holding register.

**Figure 44-11. Transmit Block Diagram**



### 44.8.3 Receive Operations

A receive frame is triggered by a start event and can be followed by synchronization data before data transmission.

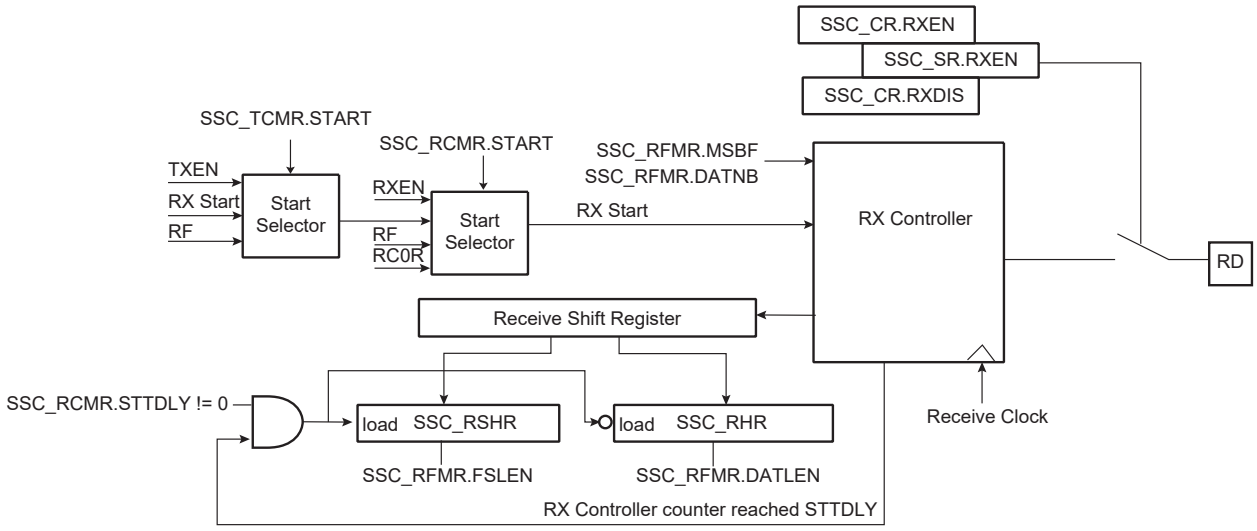
The start event is configured by setting the Receive Clock Mode Register (SSC_RCMR). See Start.

The frame synchronization is configured by setting the Receive Frame Mode Register (SSC_RFMR). See Frame Synchronization.

The receiver uses a shift register clocked by the receive clock signal and the start mode selected in the SSC_RCMR. The data is transferred from the shift register depending on the data format selected.

When the receive shift register is full, the SSC transfers the data into the Receive Holding register (SSC_RHR), the status flag RXRDY is set in the SSC_SR and the data can be read in the Receive Holding register. If another transfer occurs before read of the Receive Holding register, the status flag OVRUN is set in the SSC_SR and the receive shift register is transferred in the SSC_RHR. The old unread data is then lost.

**Figure 44-12. Receive Block Diagram**




### 44.8.4 Start

The transmitter and receiver can both be programmed to start their operations when an event occurs, respectively in the Transmit Start Selection (START) field of SSC_TCMR and in the Receive Start Selection (START) field of SSC_RCMR.

Under the following conditions the start event is independently programmable:

- Continuous. In this case, the transmission starts as soon as a word is written in SSC_THR and the reception starts as soon as the receiver is enabled.
- Synchronously with the transmitter/receiver
- On detection of a falling/rising edge on TF/RF
- On detection of a low level/high level on TF/RF
- On detection of a level change or an edge on TF/RF

A start can be programmed in the same manner on either side of the Transmit/Receive Clock Register (SSC_RCMR/SSC_TCMR). Thus, the start could be on TF (Transmit) or RF (Receive).

Moreover, the receiver can start when data is detected in the bit stream with the Compare Functions.

Detection on TF/RF input/output is done by the field FSOS of the Transmit/Receive Frame Mode Register (SSC_TFMR/SSC_RFMR).

**Figure 44-13. Transmit Start Mode**

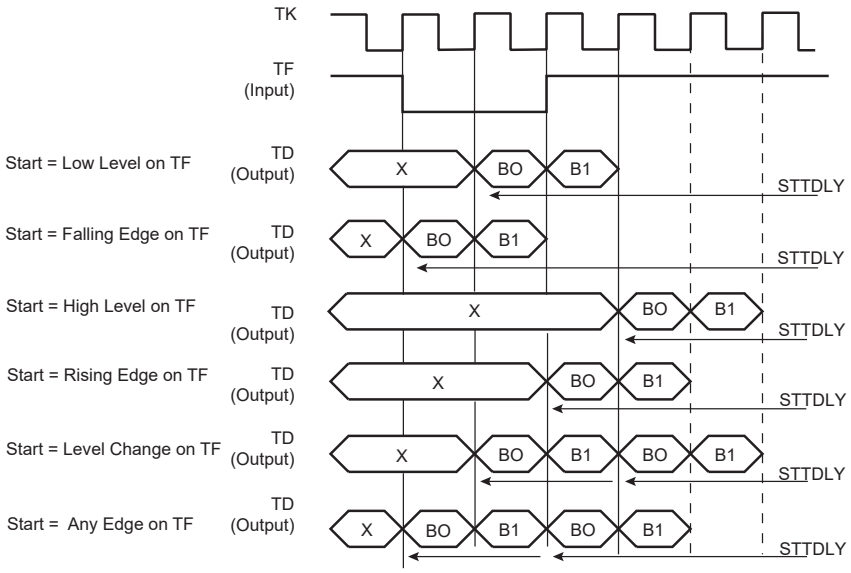
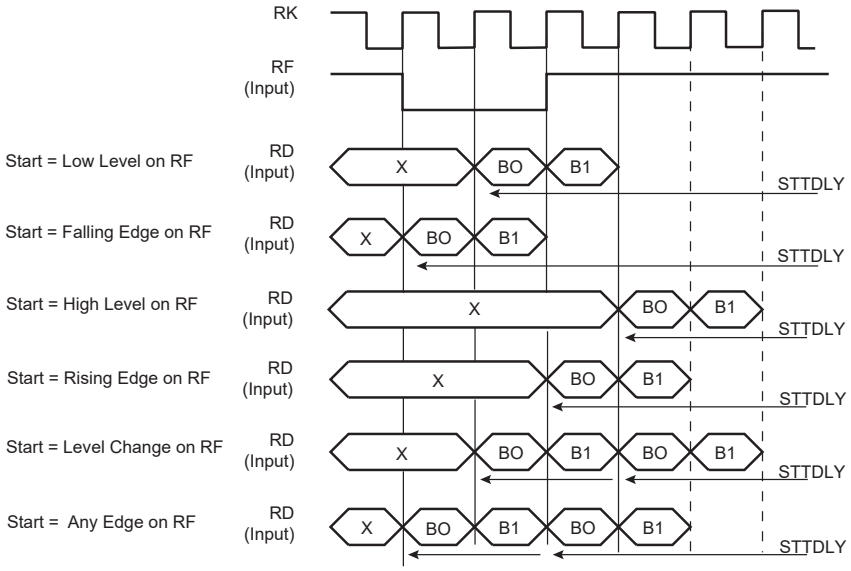



**Figure 44-14. Receive Pulse/Edge Start Modes**




### 44.8.5 Frame Synchronization

The Transmit and Receive Frame Sync pins, TF and RF, can be programmed to generate different kinds of Frame Sync signals. The Frame Sync Output Selection (FSOS) field in the Receive Frame Mode Register (SSC_RFMR) and in the Transmit Frame Mode Register (SSC_TFMR) are used to select the required waveform.

- Programmable low or high levels during data transfer are supported.
- Programmable high levels before the start of data transfers or toggling are also supported.

If a pulse waveform is selected, the Frame Sync Length (FSLEN) field in SSC_RFMR and SSC_TFMR programs the length of the pulse, from 1 bit time up to 256 bit times.

The periodicity of the Receive and Transmit Frame Sync pulse output can be programmed through the Period Divider Selection (PERIOD) field in SSC_RCMR and SSC_TCMR.

#### 44.8.5.1 Frame Sync Data

Frame Sync Data transmits or receives a specific tag during the Frame Sync signal.

During the Frame Sync signal, the receiver can sample the RD line and store the data in the Receive Sync Holding Register and the transmitter can transfer Transmit Sync Holding Register in the shift register. The data length to be sampled/shifted out during the Frame Sync signal is programmed by the FSLEN field in SSC_RFMR/SSC_TFMR and has a maximum value of 256.

Concerning the Receive Frame Sync Data operation, if the Frame Sync Length is equal to or lower than the delay between the start event and the current data reception, the data sampling operation is performed in the Receive Sync Holding Register through the receive shift register.

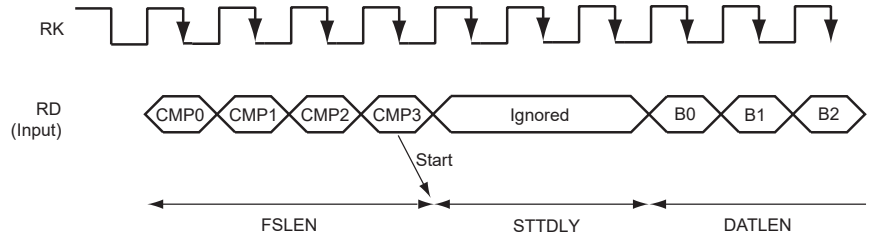
The Transmit Frame Sync Operation is performed by the transmitter only if the bit Frame Sync Data Enable (FSDEN) in SSC_TFMR is set. If the Frame Sync length is equal to or lower than the delay between the start event and the current data transmission, the normal transmission has priority and the data contained in the Transmit Sync Holding Register is transferred in the Transmit Register, then shifted out.

#### 44.8.5.2 Frame Sync Edge Detection

The Frame Sync Edge detection is programmed by the FSEDGE field in SSC_RFMR/SSC_TFMR. This sets the corresponding flags RXSYN/TXSYN in the SSC Status Register (SSC_SR) on Frame Sync Edge detection (signals RF/TF).

### 44.8.6 Receive Compare Modes

**Figure 44-15. Receive Compare Modes**




#### 44.8.6.1 Compare Functions

The length of the comparison patterns (Compare 0, Compare 1) and thus the number of bits they are compared to is defined by FSLEN, but with a maximum value of 256 bits. Comparison is always done by comparing the last bits received with the comparison pattern. Compare 0 can be one start event of the receiver. In this case, the receiver compares at each new sample the last bits received at the Compare 0 pattern contained in the Compare 0 Register (SSC_RC0R). When this start event is selected, the user can program the receiver to start a new data transfer either by writing a new Compare 0, or by receiving continuously until Compare 1 occurs. This selection is done with the STOP bit in the SSC_RCMR.

### 44.8.7 Data Format

The data framing format of both the transmitter and the receiver are programmable through the Transmitter Frame Mode Register (SSC_TFMR) and the Receive Frame Mode Register (SSC_RFMR). In either case, the user can independently select the following parameters:

- Event that starts the data transfer (START)
- Delay in number of bit periods between the start event and the first data bit (STTDLY)

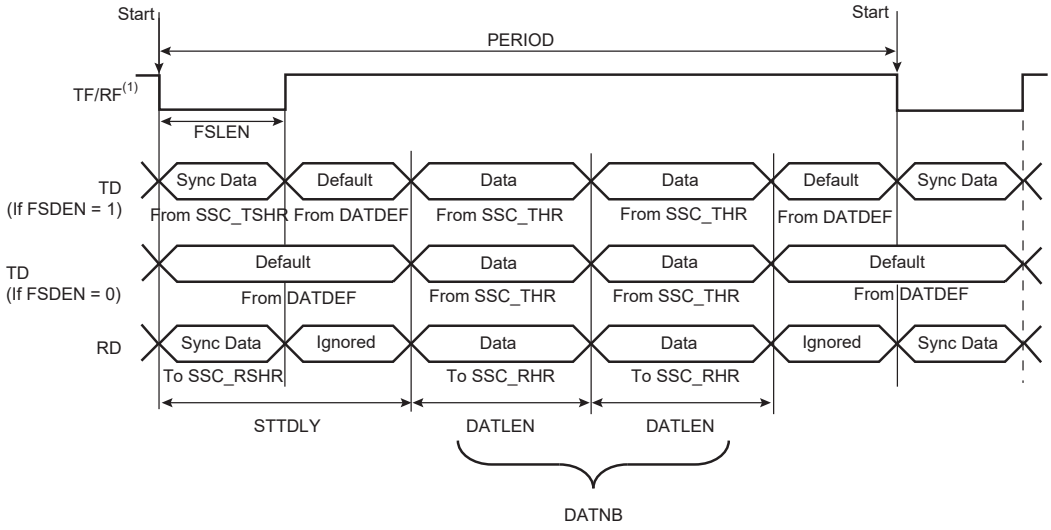
- Length of the data (DATLEN)
- Number of data to be transferred for each start event (DATNB)
- Length of synchronization transferred for each start event (FSLEN)
- Bit sense: most or least significant bit first (MSBF)

Additionally, the transmitter can be used to transfer synchronization and select the level driven on the TD pin while not in data transfer operation. This is done respectively by the Frame Sync Data Enable (FSDEN) and by the Data Default Value (DATDEF) bits in SSC_TFMR.

**Table 44-2. Data Frame Registers**

| Transmitter | Receiver | Field | Length | Comment |
|---|---|---|---|---|
| SSC_TFMR | SSC_RFMR | DATLEN | Up to 32 | Size of word |
| SSC_TFMR | SSC_RFMR | DATNB | Up to 16 | Number of words transmitted in frame |
| SSC_TFMR | SSC_RFMR | MSBF | – | Most significant bit first |
| SSC_TFMR | SSC_RFMR | FSLEN | Up to 256 | Size of Synchro data register |
| SSC_TFMR | – | DATDEF | 0 or 1 | Data default value ended |
| SSC_TFMR | – | FSDEN | – | Enable send SSC_TSHR |
| SSC_TCMR | SSC_RCMR | PERIOD | Up to 512 | Frame size |
| SSC_TCMR | SSC_RCMR | STTDLY | Up to 255 | Size of transmit start delay |

**Figure 44-16. Transmit and Receive Frame Format in Edge/Pulse Start Modes**




Note: 1. Example of input on falling edge of TF/RF.

In the example illustrated above, the SSC_THR is loaded twice. The FSDEN value has no effect on the transmission. SyncData cannot be output in Continuous mode.

**Figure 44-17. Transmit Frame Format in Continuous Mode (STTDLY = 0)**

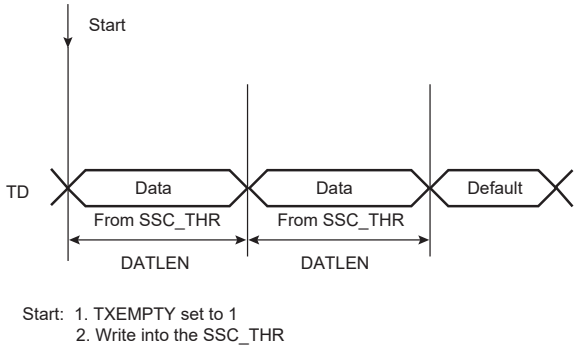
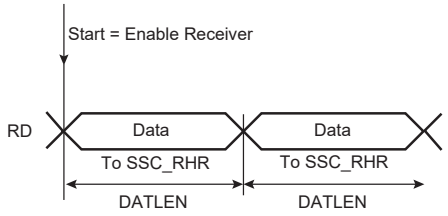



**Figure 44-18. Receive Frame Format in Continuous Mode (STTDLY = 0)**




### 44.8.8 Loop Mode

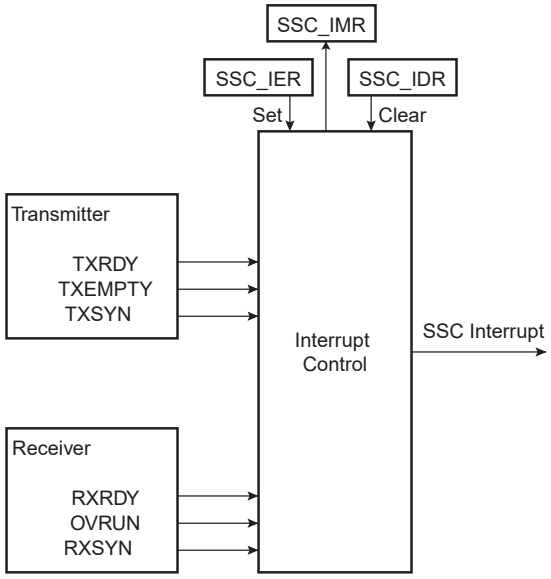
The receiver can be programmed to receive transmissions from the transmitter. This is done by setting the Loop Mode (LOOP) bit in the SSC_RFMR. In this case, RD is connected to TD, RF is connected to TF and RK is connected to TK.

### 44.8.9 Interrupt

Most bits in the SSC_SR have a corresponding bit in interrupt management registers.

The SSC can be programmed to generate an interrupt when it detects an event. The interrupt is controlled by writing the Interrupt Enable Register (SSC_IER) and Interrupt Disable Register (SSC_IDR). These registers enable and disable, respectively, the corresponding interrupt by setting and clearing the corresponding bit in the Interrupt Mask Register (SSC_IMR), which controls the generation of interrupts by asserting the SSC interrupt line connected to the interrupt controller.

**Figure 44-19. Interrupt Block Diagram**




### 44.8.10 Register Write Protection

To prevent any single software error from corrupting SSC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the SSC Write Protection Mode Register (SSC_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the SSC Write Protection Status Register (SSC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the SSC_WPSR.

The following registers can be write-protected:

- SSC Clock Mode Register
- SSC Receive Clock Mode Register
- SSC Receive Frame Mode Register
- SSC Transmit Clock Mode Register
- SSC Transmit Frame Mode Register
- SSC Receive Compare 0 Register
- SSC Receive Compare 1 Register

## 44.9 Register Summary

**Note:** Offsets 0x100–0x128 are reserved for PDC registers.

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | SSC_CR | 7:0 | | | | | | | RXDIS | RXEN |
| | | 15:8 | SWRST | | | | | | TXDIS | TXEN |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | SSC_CMR | 7:0 | DIV[7:0] | | | | | | | |
| | | 15:8 | | | | | DIV[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | SSC_RCMR | 7:0 | CKG[1:0] | | CKI | CKO[2:0] | | | CKS[1:0] | |
| | | 15:8 | | | | STOP | START[3:0] | | | |
| | | 23:16 | STTDLY[7:0] | | | | | | | |
| | | 31:24 | PERIOD[7:0] | | | | | | | |
| 0x14 | SSC_RFMR | 7:0 | MSBF | | LOOP | DATLEN[4:0] | | | | |
| | | 15:8 | | | | | DATNB[3:0] | | | |
| | | 23:16 | | FSOS[2:0] | | | FSLEN[3:0] | | | |
| | | 31:24 | FSLEN_EXT[3:0] | | | | | | | FSEDGE |
| 0x18 | SSC_TCMR | 7:0 | CKG[1:0] | | CKI | CKO[2:0] | | | CKS[1:0] | |
| | | 15:8 | | | | | START[3:0] | | | |
| | | 23:16 | STTDLY[7:0] | | | | | | | |
| | | 31:24 | PERIOD[7:0] | | | | | | | |
| 0x1C | SSC_TFMR | 7:0 | MSBF | | DATDEF | DATLEN[4:0] | | | | |
| | | 15:8 | | | | | DATNB[3:0] | | | |
| | | 23:16 | FSDEN | FSOS[2:0] | | | FSLEN[3:0] | | | |
| | | 31:24 | FSLEN_EXT[3:0] | | | | | | | FSEDGE |
| 0x20 | SSC_RHR | 7:0 | RDAT[7:0] | | | | | | | |
| | | 15:8 | RDAT[15:8] | | | | | | | |
| | | 23:16 | RDAT[23:16] | | | | | | | |
| | | 31:24 | RDAT[31:24] | | | | | | | |
| 0x24 | SSC_THR | 7:0 | TDAT[7:0] | | | | | | | |
| | | 15:8 | TDAT[15:8] | | | | | | | |
| | | 23:16 | TDAT[23:16] | | | | | | | |
| | | 31:24 | TDAT[31:24] | | | | | | | |
| 0x28 ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | SSC_RSHR | 7:0 | RSDAT[7:0] | | | | | | | |
| | | 15:8 | RSDAT[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x34 | SSC_TSHR | 7:0 | TSDAT[7:0] | | | | | | | |
| | | 15:8 | TSDAT[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x38 | SSC_RC0R | 7:0 | CP0[7:0] | | | | | | | |
| | | 15:8 | CP0[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x3C | SSC_RC1R | 7:0 | CP1[7:0] | | | | | | | |
| | | 15:8 | CP1[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x40 | SSC_SR | 7:0 | | | OVRUN | RXRDY | | | TXEMPTY | TXRDY |
| | | 15:8 | | | | | RXSYN | TXSYN | CP1 | CP0 |
| | | 23:16 | | | | | | | RXEN | TXEN |
| | | 31:24 | | | | | | | | |
| 0x44 | SSC_IER | 7:0 | | | OVRUN | RXRDY | | | TXEMPTY | TXRDY |
| | | 15:8 | | | | | RXSYN | TXSYN | CP1 | CP0 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x48 | SSC_IDR | 7:0 | | | OVRUN | RXRDY | | | TXEMPTY | TXRDY |
| | | 15:8 | | | | | RXSYN | TXSYN | CP1 | CP0 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x4C | SSC_IMR | 7:0 | | | OVRUN | RXRDY | | | TXEMPTY | TXRDY |
| | | 15:8 | | | | | RXSYN | TXSYN | CP1 | CP0 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x50 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | SSC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | SSC_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |

### 44.9.1 SSC Control Register

**Name:** SSC_CR
**Offset:** 0x0
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SWRST | | | | | | TXDIS | TXEN |
| Access | W | | | | | | W | W |
| Reset | – | | | | | | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXDIS | RXEN |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

**Bit 15 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Performs a software reset. Has priority on any other bit in SSC_CR. |

**Bit 9 – TXDIS** Transmit Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables Transmit. If a character is currently being transmitted, disables at end of current character transmission. |

**Bit 8 – TXEN** Transmit Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables Transmit if TXDIS is not set. |

**Bit 1 – RXDIS** Receive Disable

| Value | Description |
|---|---|
| 0 | No effect. |

| Value | Description |
|---|---|
| 1 | Disables Receive. If a character is currently being received, disables at end of current character reception. |

**Bit 0 – RXEN** Receive Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables Receive if RXDIS is not set. |

### 44.9.2 SSC Clock Mode Register

**Name:** SSC_CMR
**Offset:** 0x4
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | DIV[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DIV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 11:0 – DIV[11:0]** Clock Divider

| Value | Description |
|---|---|
| 0 | The Clock Divider is not active. |
| Any other value | The divided clock equals the peripheral clock divided by 2 times DIV.<br>The maximum bit rate is $f_{peripheral\ clock}/2$. The minimum bit rate is $f_{peripheral\ clock}/2 \times 4095 = f_{peripheral\ clock}/8190$. |

### 44.9.3 SSC Receive Clock Mode Register

**Name:** SSC_RCMR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PERIOD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | STTDLY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | STOP | START[3:0] | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CKG[1:0] | | CKI | CKO[2:0] | | | CKS[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – PERIOD[7:0]** Receive Period Divider Selection
This field selects the divider to apply to the selected Receive Clock in order to generate a new Frame Sync signal. If 0, no PERIOD signal is generated. If not 0, a PERIOD signal is generated each 2 x (PERIOD + 1) Receive Clock.

**Bits 23:16 – STTDLY[7:0]** Receive Start Delay
If STTDLY is not 0, a delay of STTDLY clock cycles is inserted between the start event and the current start of reception. When the receiver is programmed to start synchronously with the transmitter, the delay is also applied.

**Note:**
STTDLY must be configured in relation to the receive synchronization data to be stored in SSC_RSHR.

**Bit 12 – STOP** Receive Stop Selection

| Value | Description |
|---|---|
| 0 | After completion of a data transfer when starting with a Compare 0, the receiver stops the data transfer and waits for a new compare 0. |
| 1 | After starting a receive with a Compare 0, the receiver operates in a continuous mode until a Compare 1 is detected. |

**Bits 11:8 – START[3:0]** Receive Start Selection

| Value | Name | Description |
|---|---|---|
| 0 | CONTINUOUS | Continuous, as soon as the receiver is enabled, and immediately after the end of transfer of the previous data. |
| 1 | TRANSMIT | Transmit start |
| 2 | RF_LOW | Detection of a low level on RF signal |
| 3 | RF_HIGH | Detection of a high level on RF signal |
| 4 | RF_FALLING | Detection of a falling edge on RF signal |
| 5 | RF_RISING | Detection of a rising edge on RF signal |
| 6 | RF_LEVEL | Detection of any level change on RF signal |
| 7 | RF_EDGE | Detection of any edge on RF signal |
| 8 | CMP_0 | Compare 0 |

**Bits 7:6 – CKG[1:0]** Receive Clock Gating Selection

| Value | Name | Description |
|---|---|---|
| 0 | CONTINUOUS | None |
| 1 | EN_RF_LOW | Receive Clock enabled only if RF Low |
| 2 | EN_RF_HIGH | Receive Clock enabled only if RF High |

**Bit 5 – CKI** Receive Clock Inversion
CKI affects only the Receive Clock and not the output clock signal.

| Value | Description |
|---|---|
| 0 | The data inputs (Data and Frame Sync signals) are sampled on Receive Clock falling edge. The Frame Sync signal output is shifted out on Receive Clock rising edge. |
| 1 | The data inputs (Data and Frame Sync signals) are sampled on Receive Clock rising edge. The Frame Sync signal output is shifted out on Receive Clock falling edge. |

**Bits 4:2 – CKO[2:0]** Receive Clock Output Mode Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None, RK pin is an input |
| 1 | CONTINUOUS | Continuous Receive Clock, RK pin is an output |
| 2 | TRANSFER | Receive Clock only during data transfers, RK pin is an output |

**Bits 1:0 – CKS[1:0]** Receive Clock Selection

| Value | Name | Description |
|---|---|---|
| 0 | MCK | Divided Clock |
| 1 | TK | TK Clock signal |
| 2 | RK | RK pin |

### 44.9.4 SSC Receive Frame Mode Register

**Name:** SSC_RFMR
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FSLEN_EXT[3:0] | | | | | | | FSEDGE |
| Access | R/W | R/W | R/W | R/W | | | | R/W |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | FSOS[2:0] | | | FSLEN[3:0] | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | DATNB[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MSBF | | LOOP | DATLEN[4:0] | | | | |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:28 – FSLEN_EXT[3:0]** FSLEN Field Extension
Extends FSLEN field. For details, see FSLEN: Receive Frame Sync Length.

**Bit 24 – FSEDGE** Frame Sync Edge Detection
Determines which edge on Frame Sync will generate the interrupt RXSYN in the SSC Status Register.

| Value | Name | Description |
|---|---|---|
| 0 | POSITIVE | Positive Edge Detection |
| 1 | NEGATIVE | Negative Edge Detection |

**Bits 22:20 – FSOS[2:0]** Receive Frame Sync Output Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None, RF pin is an input |
| 1 | NEGATIVE | Negative Pulse, RF pin is an output |
| 2 | POSITIVE | Positive Pulse, RF pin is an output |
| 3 | LOW | Driven Low during data transfer, RF pin is an output |
| 4 | HIGH | Driven High during data transfer, RF pin is an output |
| 5 | TOGGLING | Toggling at each start of data transfer, RF pin is an output |

**Bits 19:16 – FSLEN[3:0]** Receive Frame Sync Length
This field defines the number of bits sampled and stored in the Receive Sync Data Register. When this mode is selected by the START field in the Receive Clock Mode Register, it also determines the length of the sampled data to be compared to the Compare 0 or Compare 1 register.

This field is used with FSLEN_EXT to determine the pulse length of the Receive Frame Sync signal.

Pulse length is equal to FSLEN + (FSLEN_EXT × 16) + 1 Receive Clock periods.

**Bits 11:8 – DATNB[3:0]** Data Number per Frame
This field defines the number of data words to be received after each transfer start, which is equal to (DATNB + 1).

**Bit 7 – MSBF** Most Significant Bit First

| Value | Description |
|---|---|
| 0 | The lowest significant bit of the data register is sampled first in the bit stream. |
| 1 | The most significant bit of the data register is sampled first in the bit stream. |

**Bit 5 – LOOP** Loop Mode

| Value | Description |
|---|---|
| 0 | Normal operating mode. |
| 1 | RD is driven by TD, RF is driven by TF and TK drives RK. |

**Bits 4:0 – DATLEN[4:0]** Data Length

| Value | Description |
|---|---|
| 0 | Forbidden value (1-bit data length not supported). |
| Any other value | The bit stream contains DATLEN + 1 data bits. |

### 44.9.5 SSC Transmit Clock Mode Register

**Name:** SSC_TCMR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PERIOD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | STTDLY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | START[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CKG[1:0] | | CKI | CKO[2:0] | | | CKS[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – PERIOD[7:0]** Transmit Period Divider Selection
This field selects the divider to apply to the selected Transmit Clock to generate a new Frame Sync signal. If 0, no period signal is generated. If not 0, a period signal is generated at each 2 × (PERIOD + 1) Transmit Clock.

**Bits 23:16 – STTDLY[7:0]** Transmit Start Delay
If STTDLY is not 0, a delay of STTDLY clock cycles is inserted between the start event and the current start of transmission of data. When the transmitter is programmed to start synchronously with the receiver, the delay is also applied.

**Note:**
If STTDLY is too short with respect to transmit synchronization data (SSC_TSHR), SSC_THR.TDAT is transmitted instead of the end of SSC_TSHR.

**Bits 11:8 – START[3:0]** Transmit Start Selection

| Value | Name | Description |
|---|---|---|
| 0 | CONTINUOUS | Continuous, as soon as a word is written in the SSC_THR (if Transmit is enabled), and immediately after the end of transfer of the previous data |
| 1 | RECEIVE | Receive start |
| 2 | TF_LOW | Detection of a low level on TF signal |

| Value | Name | Description |
|---|---|---|
| 3 | TF_HIGH | Detection of a high level on TF signal |
| 4 | TF_FALLING | Detection of a falling edge on TF signal |
| 5 | TF_RISING | Detection of a rising edge on TF signal |
| 6 | TF_LEVEL | Detection of any level change on TF signal |
| 7 | TF_EDGE | Detection of any edge on TF signal |

**Bits 7:6 – CKG[1:0]** Transmit Clock Gating Selection

| Value | Name | Description |
|---|---|---|
| 0 | CONTINUOUS | None |
| 1 | EN_TF_LOW | Transmit Clock enabled only if TF Low |
| 2 | EN_TF_HIGH | Transmit Clock enabled only if TF High |

**Bit 5 – CKI** Transmit Clock Inversion
CKI affects only the Transmit Clock and not the Output Clock signal.

| Value | Description |
|---|---|
| 0 | The data outputs (Data and Frame Sync signals) are shifted out on Transmit Clock falling edge. The Frame Sync signal input is sampled on Transmit Clock rising edge. |
| 1 | The data outputs (Data and Frame Sync signals) are shifted out on Transmit Clock rising edge. The Frame Sync signal input is sampled on Transmit Clock falling edge. |

**Bits 4:2 – CKO[2:0]** Transmit Clock Output Mode Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None, TK pin is an input |
| 1 | CONTINUOUS | Continuous Transmit Clock, TK pin is an output |
| 2 | TRANSFER | Transmit Clock only during data transfers, TK pin is an output |

**Bits 1:0 – CKS[1:0]** Transmit Clock Selection

| Value | Name | Description |
|---|---|---|
| 0 | MCK | Divided Clock |
| 1 | RK | RK Clock signal |
| 2 | TK | TK pin |

### 44.9.6 SSC Transmit Frame Mode Register

**Name:** SSC_TFMR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FSLEN_EXT[3:0] | | | | | | | FSEDGE |
| Access | R/W | R/W | R/W | R/W | | | | R/W |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FSDEN | FSOS[2:0] | | | FSLEN[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | DATNB[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MSBF | | DATDEF | DATLEN[4:0] | | | | |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:28 – FSLEN_EXT[3:0]** FSLEN Field Extension
Extends FSLEN field. For details, seee FSLEN bit description below.

**Bit 24 – FSEDGE** Frame Sync Edge Detection
Determines which edge on frame synchronization will generate the interrupt TXSYN (Status Register).

| Value | Name | Description |
|---|---|---|
| 0 | POSITIVE | Positive Edge Detection |
| 1 | NEGATIVE | Negative Edge Detection |

**Bit 23 – FSDEN** Frame Sync Data Enable

| Value | Description |
|---|---|
| 0 | The TD line is driven with the default value during the Transmit Frame Sync signal. |
| 1 | SSC_TSHR value is shifted out during the transmission of the Transmit Frame Sync signal. |

**Bits 22:20 – FSOS[2:0]** Transmit Frame Sync Output Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None, TF pin is an input |
| 1 | NEGATIVE | Negative Pulse, TF pin is an output |

| Value | Name | Description |
|---|---|---|
| 2 | POSITIVE | Positive Pulse, TF pin is an output |
| 3 | LOW | Driven Low during data transfer |
| 4 | HIGH | Driven High during data transfer |
| 5 | TOGGLING | Toggling at each start of data transfer |

**Bits 19:16 – FSLEN[3:0]** Transmit Frame Sync Length
This field defines the length of the Transmit Frame Sync signal and the number of bits shifted out from SSC_TSHR if FSDEN is 1.

This field is used with FSLEN_EXT to determine the pulse length of the Transmit Frame Sync signal.

Pulse length is equal to FSLEN + (FSLEN_EXT × 16) + 1 Transmit Clock period.

**Bits 11:8 – DATNB[3:0]** Data Number per Frame
This field defines the number of data words to be transferred after each transfer start, which is equal to (DATNB + 1).

**Bit 7 – MSBF** Most Significant Bit First

| Value | Description |
|---|---|
| 0 | The lowest significant bit of the data register is shifted out first in the bit stream. |
| 1 | The most significant bit of the data register is shifted out first in the bit stream. |

**Bit 5 – DATDEF** Data Default Value
This bit defines the level driven on the TD pin while out of transmission. Note that if the pin is defined as multi-drive by the PIO Controller, the pin is enabled only if the SCC TD output is 1.

**Bits 4:0 – DATLEN[4:0]** Data Length

| Value | Description |
|---|---|
| 0 | Forbidden value (1-bit data length not supported). |
| Any other value | The bit stream contains DATLEN + 1 data bits. |

### 44.9.7 SSC Receive Holding Register

**Name:** SSC_RHR
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RDAT[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RDAT[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RDAT[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RDAT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RDAT[31:0]** Receive Data
Right aligned regardless of the number of data bits defined by DATLEN in SSC_RFMR.

### 44.9.8 SSC Transmit Holding Register

**Name:** SSC_THR
**Offset:** 0x24
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TDAT[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TDAT[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TDAT[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TDAT[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – TDAT[31:0]** Transmit Data
Right aligned regardless of the number of data bits defined by DATLEN in SSC_TFMR.

### 44.9.9 SSC Receive Synchronization Holding Register

**Name:** SSC_RSHR
**Offset:** 0x30
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RSDAT[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RSDAT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RSDAT[15:0]** Receive Synchronization Data

### 44.9.10 SSC Transmit Synchronization Holding Register

**Name:** SSC_TSHR
**Offset:** 0x34
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TSDAT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TSDAT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – TSDAT[15:0]** Transmit Synchronization Data

### 44.9.11 SSC Receive Compare 0 Register

**Name:** SSC_RC0R
**Offset:** 0x38
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CP0[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CP0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – CP0[15:0]** Receive Compare Data 0

### 44.9.12 SSC Receive Compare 1 Register

**Name:** SSC_RC1R
**Offset:** 0x3C
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CP1[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CP1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – CP1[15:0]** Receive Compare Data 1

### 44.9.13 SSC Status Register

**Name:** SSC_SR
**Offset:** 0x40
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXEN | TXEN |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RXSYN | TXSYN | CP1 | CP0 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | OVRUN | RXRDY | | | TXEMPTY | TXRDY |
| Access | | | R | R | | | R | R |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bit 17 – RXEN** Receive Enable

| Value | Description |
|---|---|
| 0 | Receive is disabled. |
| 1 | Receive is enabled. |

**Bit 16 – TXEN** Transmit Enable

| Value | Description |
|---|---|
| 0 | Transmit is disabled. |
| 1 | Transmit is enabled. |

**Bit 11 – RXSYN** Receive Sync

| Value | Description |
|---|---|
| 0 | An Rx Sync has not occurred since the last read of the Status Register. |
| 1 | An Rx Sync has occurred since the last read of the Status Register. |

**Bit 10 – TXSYN** Transmit Sync

| Value | Description |
|---|---|
| 0 | A Tx Sync has not occurred since the last read of the Status Register. |
| 1 | A Tx Sync has occurred since the last read of the Status Register. |

**Bit 9 – CP1** Compare 1

| Value | Description |
|---|---|
| 0 | A compare 1 has not occurred since the last read of the Status Register. |
| 1 | A compare 1 has occurred since the last read of the Status Register. |

**Bit 8 – CP0** Compare 0

| Value | Description |
|---|---|
| 0 | A compare 0 has not occurred since the last read of the Status Register. |
| 1 | A compare 0 has occurred since the last read of the Status Register. |

**Bit 5 – OVRUN** Receive Overrun

| Value | Description |
|---|---|
| 0 | No data has been loaded in SSC_RHR while previous data has not been read since the last read of the Status Register. |
| 1 | Data has been loaded in SSC_RHR while previous data has not yet been read since the last read of the Status Register. |

**Bit 4 – RXRDY** Receive Ready

| Value | Description |
|---|---|
| 0 | SSC_RHR is empty. |
| 1 | Data has been received and loaded in SSC_RHR. |

**Bit 1 – TXEMPTY** Transmit Empty

| Value | Description |
|---|---|
| 0 | Data remains in SSC_THR or is currently transmitted from TSR. |
| 1 | Last data written in SSC_THR has been loaded in TSR and last data loaded in TSR has been transmitted. |

**Bit 0 – TXRDY** Transmit Ready

| Value | Description |
|---|---|
| 0 | Data has been loaded in SSC_THR and is waiting to be loaded in the transmit shift register (TSR). |
| 1 | SSC_THR is empty. |

### 44.9.14 SSC Interrupt Enable Register

**Name:** SSC_IER
**Offset:** 0x44
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RXSYN | TXSYN | CP1 | CP0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | OVRUN | RXRDY | | | TXEMPTY | TXRDY |
| Access | | | W | W | | | W | W |
| Reset | | | – | – | | | – | – |

**Bit 11 – RXSYN** Rx Sync Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Rx Sync Interrupt. |

**Bit 10 – TXSYN** Tx Sync Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Tx Sync Interrupt. |

**Bit 9 – CP1** Compare 1 Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Compare 1 Interrupt. |

**Bit 8 – CP0** Compare 0 Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Compare 0 Interrupt. |

**Bit 5 – OVRUN** Receive Overrun Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Receive Overrun Interrupt. |

**Bit 4 – RXRDY** Receive Ready Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Receive Ready Interrupt. |

**Bit 1 – TXEMPTY** Transmit Empty Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Transmit Empty Interrupt. |

**Bit 0 – TXRDY** Transmit Ready Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Transmit Ready Interrupt. |

### 44.9.15 SSC Interrupt Disable Register

**Name:** SSC_IDR
**Offset:** 0x48
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RXSYN | TXSYN | CP1 | CP0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | OVRUN | RXRDY | | | TXEMPTY | TXRDY |
| Access | | | W | W | | | W | W |
| Reset | | | – | – | | | – | – |

**Bit 11 – RXSYN** Rx Sync Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Rx Sync Interrupt. |

**Bit 10 – TXSYN** Tx Sync Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Tx Sync Interrupt. |

**Bit 9 – CP1** Compare 1 Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Compare 1 Interrupt. |

**Bit 8 – CP0** Compare 0 Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Compare 0 Interrupt. |

**Bit 5 – OVRUN** Receive Overrun Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Receive Overrun Interrupt. |

**Bit 4 – RXRDY** Receive Ready Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Receive Ready Interrupt. |

**Bit 1 – TXEMPTY** Transmit Empty Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Transmit Empty Interrupt. |

**Bit 0 – TXRDY** Transmit Ready Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Transmit Ready Interrupt. |

### 44.9.16 SSC Interrupt Mask Register

**Name:** SSC_IMR
**Offset:** 0x4C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RXSYN | TXSYN | CP1 | CP0 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | OVRUN | RXRDY | | | TXEMPTY | TXRDY |
| Access | | | R | R | | | R | R |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bit 11 – RXSYN** Rx Sync Interrupt Mask

| Value | Description |
|---|---|
| 0 | The Rx Sync Interrupt is disabled. |
| 1 | The Rx Sync Interrupt is enabled. |

**Bit 10 – TXSYN** Tx Sync Interrupt Mask

| Value | Description |
|---|---|
| 0 | The Tx Sync Interrupt is disabled. |
| 1 | The Tx Sync Interrupt is enabled. |

**Bit 9 – CP1** Compare 1 Interrupt Mask

| Value | Description |
|---|---|
| 0 | The Compare 1 Interrupt is disabled. |
| 1 | The Compare 1 Interrupt is enabled. |

**Bit 8 – CP0** Compare 0 Interrupt Mask

| Value | Description |
|---|---|
| 0 | The Compare 0 Interrupt is disabled. |
| 1 | The Compare 0 Interrupt is enabled. |

**Bit 5 – OVRUN** Receive Overrun Interrupt Mask

| Value | Description |
|---|---|
| 0 | The Receive Overrun Interrupt is disabled. |
| 1 | The Receive Overrun Interrupt is enabled. |

**Bit 4 – RXRDY** Receive Ready Interrupt Mask

| Value | Description |
|---|---|
| 0 | The Receive Ready Interrupt is disabled. |
| 1 | The Receive Ready Interrupt is enabled. |

**Bit 1 – TXEMPTY** Transmit Empty Interrupt Mask

| Value | Description |
|---|---|
| 0 | The Transmit Empty Interrupt is disabled. |
| 1 | The Transmit Empty Interrupt is enabled. |

**Bit 0 – TXRDY** Transmit Ready Interrupt Mask

| Value | Description |
|---|---|
| 0 | The Transmit Ready Interrupt is disabled. |
| 1 | The Transmit Ready Interrupt is enabled. |

### 44.9.17 SSC Write Protection Mode Register

**Name:** SSC_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x535343 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See Register Write Protection for the list of registers that can be protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x535343 (“SSC” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x535343 (“SSC” in ASCII). |

### 44.9.18 SSC Write Protection Status Register

**Name:** SSC_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protect Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of the SSC_WPSR. |
| 1 | A write protection violation has occurred since the last read of the SSC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 45. Two-wire Interface (TWIHS)

## 45.1 Description

The Two-wire Interface (TWIHS) interconnects components on a unique two-wire bus, made up of one clock line and one data line with speeds of up to 400 kbit/s in Fast mode and up to 3.4 Mbit/s in High-speed slave mode only, based on a byte-oriented transfer format. It can be used with any Two-wire Interface bus Serial EEPROM and I²C-compatible devices, such as a Real-Time Clock (RTC), Dot Matrix/Graphic LCD Controller and temperature sensor. The TWIHS is programmable as a master or a slave with sequential or single-byte access. Multiple master capability is supported.

A configurable baud rate generator permits the output data rate to be adapted to a wide range of core clock frequencies.

The table below lists the compatibility level of the Two-wire Interface in Master mode and a full $I^2C$ compatible device.

**Table 45-1. TWI Compatibility with $I^2C$ Standard**

| $I^2C$ Standard | TWI |
|---|---|
| Standard Mode Speed (100 kHz) | Supported |
| Fast Mode Speed (400 kHz) | Supported |
| High-speed Mode (Slave only, 3.4 MHz) | Supported |
| 7- or 10-bit(1) Slave Addressing | Supported |
| START Byte(2) | Not Supported |
| Repeated Start (Sr) Condition | Supported |
| ACK and NACK Management | Supported |
| Input Filtering | Supported |
| Slope Control | Not Supported |
| Clock Stretching | Supported |
| Multi Master Capability | Supported |

**Note:**

1. 10-bit support in Master mode only.
2. START + 00000001 + Ack + Sr

## 45.2 Embedded Characteristics

- 2 TWIHSs
- 16-byte Transmit and Receive FIFOs
- Compatible with Two-wire Interface Serial Memory and I²C Compatible Devices(1)
- One, Two or Three Bytes for Slave Address
- Sequential Read/Write Operations

- Master and Multimaster Operation (Standard and Fast Modes Only)
- Slave Mode Operation (Standard, Fast and High-Speed Modes)
- Bit Rate: Up to 400 Kbit/s in Fast Mode and 3.4 Mbit/s in High-Speed Mode (Slave Mode Only)
- General Call Supported in Slave Mode
- SleepWalking (Asynchronous and Partial Wakeup)
- SMBus Support
- Connection to DMA Controller (DMA) Channel Capabilities Optimizes Data Transfers
- Register Write Protection

**Note:**
See TWI Compatibility with I2C Standard for details on compatibility with I²C Standard.

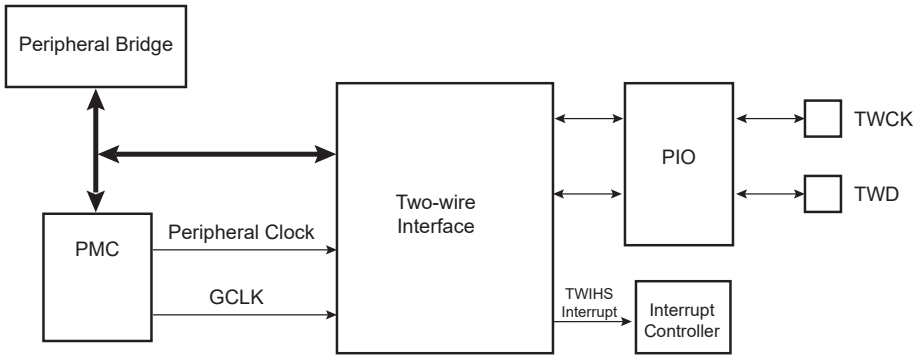
## 45.3 List of Abbreviations

**Table 45-2. Abbreviations**

| Abbreviation | Description |
|---|---|
| TWI | Two-wire Interface |
| A | Acknowledge |
| NA | Non Acknowledge |
| P | Stop |
| S | Start |
| Sr | Repeated Start |
| SADR | Slave Address |
| ADR | Any address except SADR |
| R | Read |
| W | Write |

## 45.4 Block Diagram

**Figure 45-1. Block Diagram**



### 45.4.1 I/O Lines Description

**Table 45-3. I/O Lines Description**

| Pin Name | Pin Description | Type |
|---|---|---|
| TWD | Two-wire Serial Data | Input/Output |
| TWCK | Two-wire Serial Clock | Input/Output |

## 45.5 Product Dependencies

### 45.5.1 I/O Lines

Both TWD and TWCK are bidirectional lines, connected to a positive supply voltage via a current source or pullup resistor. When the bus is free, both lines are high. The output stages of devices connected to the bus must have an open-drain or open-collector to perform the wired-AND function.

TWD and TWCK pins may be multiplexed with PIO lines. To enable the TWIHS, the user must program the PIO Controller to dedicate TWD and TWCK as peripheral lines. When High-speed Slave mode is enabled, the analog pad filter must be enabled.

The user must not program TWD and TWCK as open-drain. This is already done by the hardware.

### 45.5.2 Power Management

Enable the peripheral clock.

The TWIHS may be clocked through the Power Management Controller (PMC), thus the user must first configure the PMC to enable the TWIHS clock.

### 45.5.3 Interrupt Sources

The TWIHS has an interrupt line connected to the Interrupt Controller. In order to handle interrupts, the Interrupt Controller must be programmed before configuring the TWIHS.

## 45.6 Functional Description

### 45.6.1 Transfer Format

The data put on the TWD line must be 8 bits long. Data is transferred MSB first; each byte must be followed by an acknowledgement. The number of bytes per transfer is unlimited, shown in Transfer Format.

Each transfer begins with a START condition and terminates with a STOP condition, as shown in START and STOP Conditions.

- A high-to-low transition on the TWD line while TWCK is high defines the START condition.
- A low-to-high transition on the TWD line while TWCK is high defines the STOP condition.

**Figure 45-2. START and STOP Conditions**

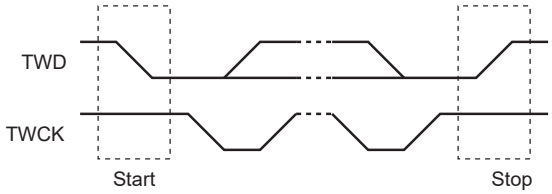
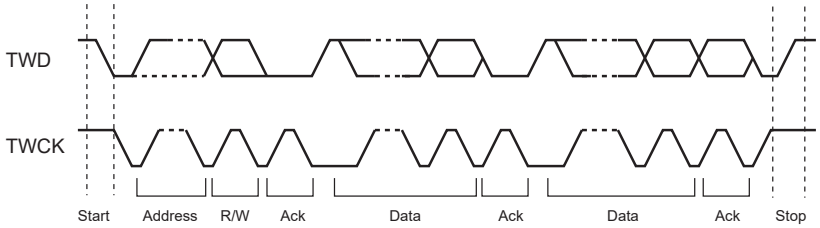


**Figure 45-3. Transfer Format**




### 45.6.2 Modes of Operation

The TWIHS has different modes of operation:

- Master Transmitter mode (Standard and Fast modes only)
- Master Receiver mode (Standard and Fast modes only)
- Multimaster Transmitter mode (Standard and Fast modes only)
- Multimaster Receiver mode (Standard and Fast modes only)
- Slave Transmitter mode (Standard, Fast and High-speed modes)
- Slave Receiver mode (Standard, Fast and High-speed modes)

These modes are described in the following sections.

### 45.6.3 Master Mode

#### 45.6.3.1 Definition

The master is the device that starts a transfer, generates a clock and stops it. This operating mode is not available if High-speed mode is selected.

#### 45.6.3.2 Programming Master Mode

The following registers must be programmed before entering Master mode:

1. TWIHS_MMR.DADR (+ IADRSZ + IADR if a 10-bit device is addressed): The device address is used to access slave devices in Read or Write mode.
2. TWIHS_CWGR.CKDIV + CHDIV + CLDIV: Clock Waveform register
3. TWIHS_CR.SVDIS: Disables the Slave mode
4. TWIHS_CR.MSEN: Enables the Master mode

**Note:** If the TWIHS is already in Master mode, the device address (DADR) can be configured without disabling the Master mode.

#### 45.6.3.3 Transfer Rate Clock Source

The TWIHS speed is defined in the TWIHS_CWGR. The TWIHS baud rate can be based either on the peripheral clock if the CKSRC bit value is ‘0’ or on a GCLK clock if the CKSRC bit value is ‘1’.

If CKSRC = 1, the baud rate is independent of the system/core clock (MCK) and thus the MCK frequency can be changed without affecting the TWIHS transfer rate.

The GCLK frequency must always be three times lower than the peripheral clock frequency.

#### 45.6.3.4 Master Transmitter Mode

This operating mode is not available if High-speed mode is selected.

After the master initiates a START condition when writing into the Transmit Holding register (TWIHS_THR), it sends a 7-bit slave address, configured in the Master Mode register (DADR in TWIHS_MMR), to notify the slave device. The bit following the slave address indicates the transfer direction, 0 in this case (MREAD = 0 in TWIHS_MMR).

The TWIHS transfers require the slave to acknowledge each received byte. During the acknowledge clock pulse (9th pulse), the master releases the data line (HIGH), enabling the slave to pull it down in order to generate the acknowledge. If the slave does not acknowledge the byte, then the Not Acknowledge flag (NACK) is set in the TWIHS Status Register (TWIHS_SR) of the master and a STOP condition is sent. The NACK flag must be cleared by reading TWIHS_SR before the next write into TWIHS_THR. As with the other status bits, an interrupt can be generated if enabled in the Interrupt Enable register (TWIHS_IER). If the slave acknowledges the byte, the data written in the TWIHS_THR is then shifted in the internal shifter and transferred. When an acknowledge is detected, the TXRDY bit is set until a new write in the TWIHS_THR.

TXRDY is used as Transmit Ready for the DMA transmit channel.

While no new data is written in the TWIHS_THR, the serial clock line is tied low. When new data is written in the TWIHS_THR, the SCL is released and the data is sent. Setting the STOP bit in TWIHS_CR generates a STOP condition.

After a master write transfer, the serial clock line is stretched (tied low) as long as no new data is written in the TWIHS_THR or until a STOP command is performed.

To clear the TXRDY flag, first set the bit TWIHS_CR.MSDIS, then set the bit TWIHS_CR.MSEN.

See the figures below.

**Figure 45-4. Master Write with One Data Byte**

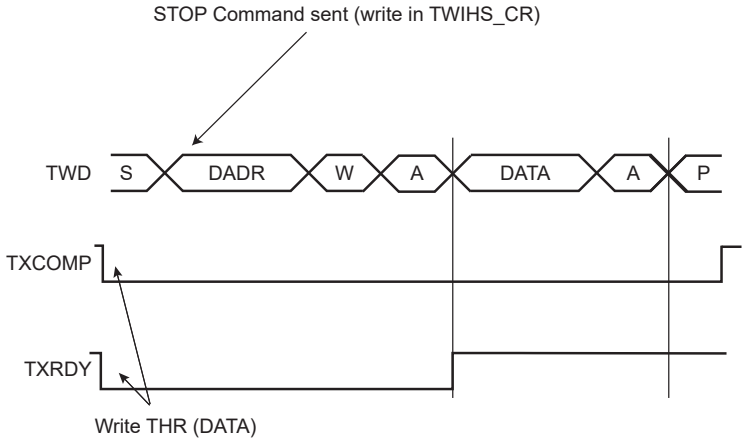


**Figure 45-5. Master Write with Multiple Data Bytes**

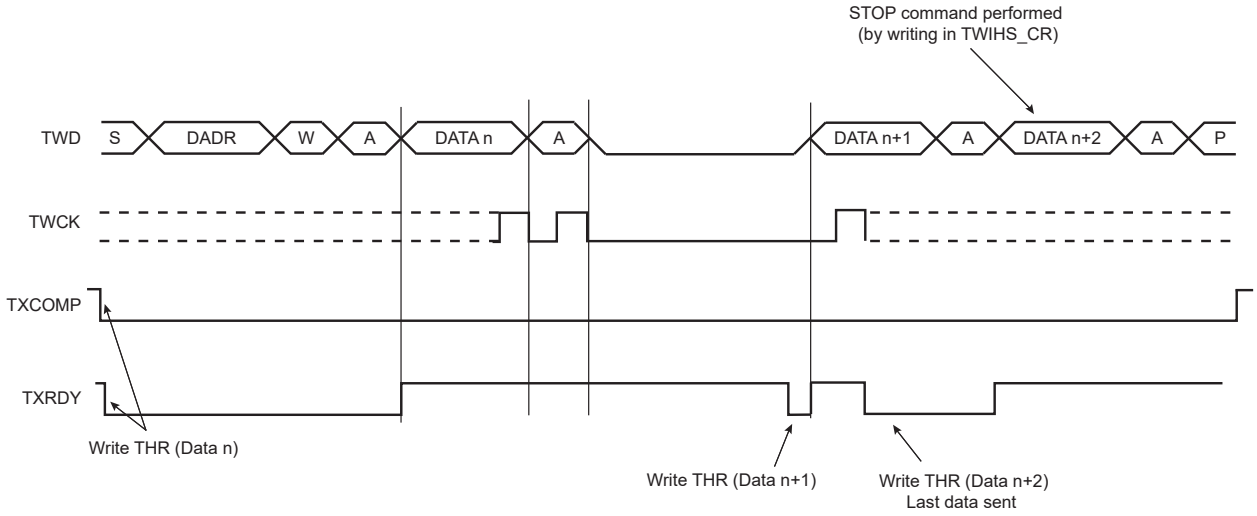
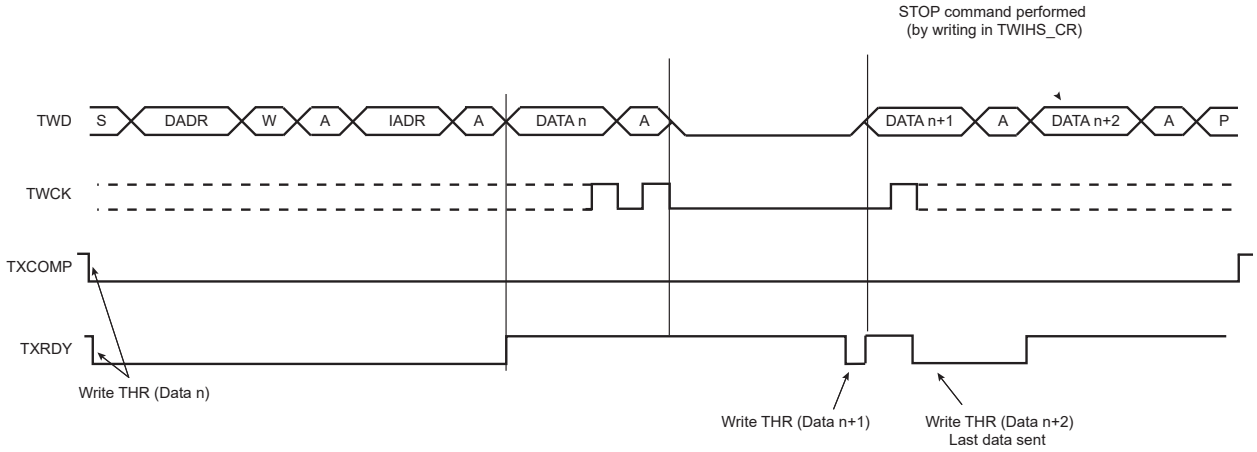



**Figure 45-6. Master Write with One-Byte Internal Address and Multiple Data Bytes**




### 45.6.3.5 Master Receiver Mode

Master Receiver mode is not available if High-speed mode is selected.

The read sequence begins by setting the START bit. After the START condition has been sent, the master sends a 7-bit slave address to notify the slave device. The bit following the slave address indicates the transfer direction, 1 in this case (MREAD = 1 in TWIHS_MMR). During the acknowledge clock pulse (9th pulse), the master releases the data line (HIGH), enabling the slave to pull it down in order to generate the acknowledge. The master polls the data line during this clock pulse and sets TWIHS_SR.NACK if the slave does not acknowledge the byte.

If an acknowledge is received, the master is then ready to receive data from the slave. After data has been received, the master sends an acknowledge condition to notify the slave that the data has been received except for the last data (see Master Read with One Data Byte). When TWIHS_SR.RXRDY is set, a character has been received in the Receive Holding register (TWIHS_RHR). The RXRDY bit is reset when reading the TWIHS_RHR.

When a single data byte read is performed, with or without internal address (IADR), the START and STOP bits must be set at the same time. See Master Read with One Data Byte. When a multiple data byte read is performed, with or without internal address (IADR), the STOP bit must be set after the next-

to-last data received (same condition applies for START bit to generate a REPEATED START). See Master Read with Multiple Data Bytes. For internal address usage, see Internal Address.

If TWIHS_RHR is full (RXRDY high) and the master is receiving data, the serial clock line is tied low before receiving the last bit of the data and until the TWIHS_RHR is read. Once the TWIHS_RHR is read, the master stops stretching the serial clock line and ends the data reception. See Master Read Clock Stretching with Multiple Data Bytes.

**WARNING** When receiving multiple bytes in Master Read mode, if the next-to-last access is not read (the RXRDY flag remains high), the last access is not completed until TWIHS_RHR is read. The last access stops on the next-to-last bit (clock stretching). When the TWIHS_RHR is read, there is only half a bit period to send the STOP (or START) command, else another read access might occur (spurious access).

A possible workaround is to set the STOP (or START) bit before reading the TWIHS_RHR on the next-to-last access (within IT handler).

**Figure 45-7. Master Read with One Data Byte**

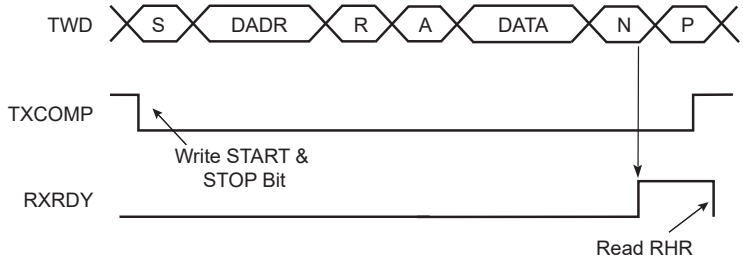



**Figure 45-8. Master Read with Multiple Data Bytes**

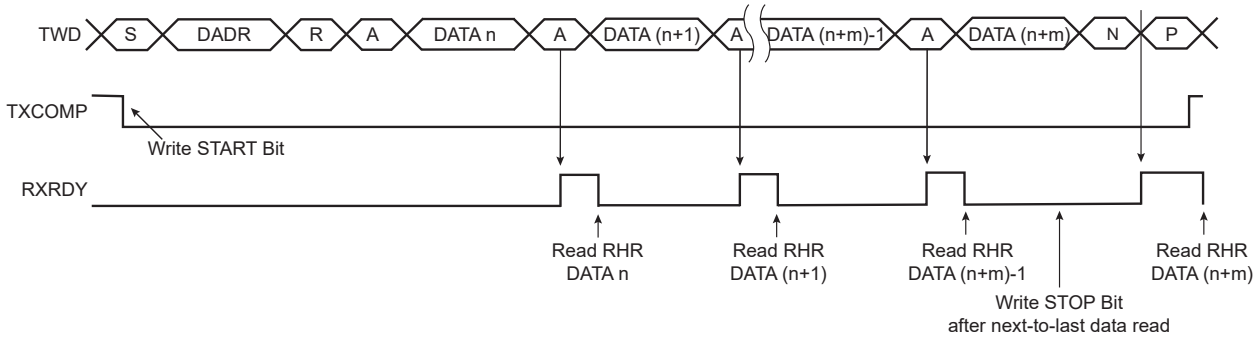
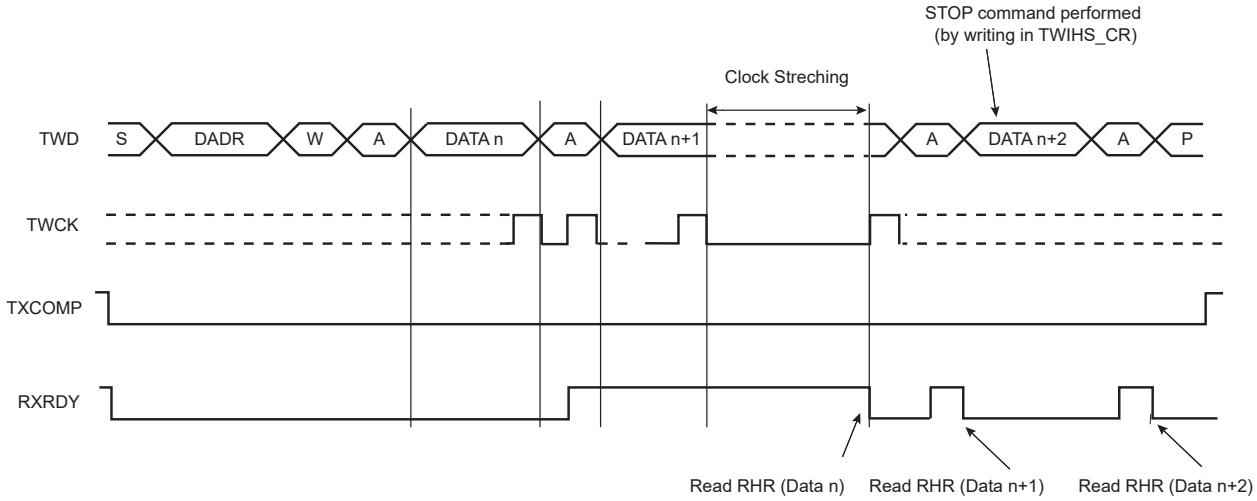


**Figure 45-9. Master Read Clock Stretching with Multiple Data Bytes**




RXRDY is used as receive ready for the DMA receive channel.

### 45.6.3.6 Internal Address

The TWIHS can perform transfers with 7-bit slave address devices and with 10-bit slave address devices.

#### 45.6.3.6.1 7-bit Slave Addressing

When addressing 7-bit slave devices, the internal address bytes are used to perform random address (read or write) accesses to reach one or more data bytes, e.g. within a memory page location in a serial memory. When performing read operations with an internal address, the TWIHS performs a write operation to set the internal address into the slave device, and then switch to Master Receiver mode. Note that the second START condition (after sending the IADR) is sometimes called “repeated start” (Sr) in $I^2C$ fully-compatible devices. See Master Read with One-, Two- or Three-Byte Internal Address and One Data Byte.

See Master Write with One-, Two- or Three-Byte Internal Address and One Data Byte and Internal Address Usage for the master write operation with internal address.

The three internal address bytes are configurable through TWIHS_MMR.

If the slave device supports only a 7-bit address, i.e., no internal address, IADRSZ must be set to 0.

The table below shows the abbreviations used in the figures below.

**Table 45-4. Abbreviations**

| Abbreviation | Definition |
|---|---|
| S | Start |
| Sr | Repeated Start |
| P | Stop |
| W | Write |
| R | Read |
| A | Acknowledge |
| NA | Not Acknowledge |
| DADR | Device Address |

| .....continued | |
|---|---|
| **Abbreviation** | **Definition** |
| IADR | Internal Address |

**Figure 45-10. Master Write with One-, Two- or Three-Byte Internal Address and One Data Byte**

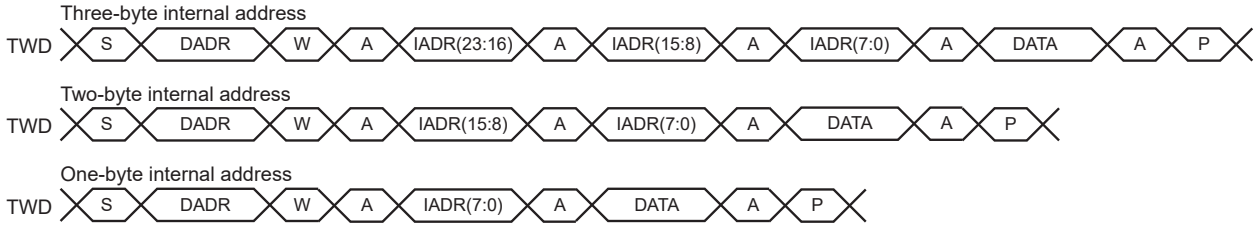
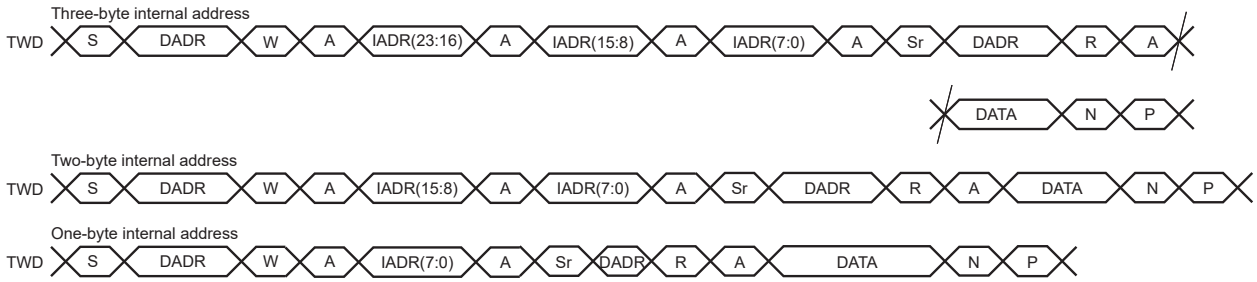



**Figure 45-11. Master Read with One-, Two- or Three-Byte Internal Address and One Data Byte**




#### 45.6.3.6.2 10-bit Slave Addressing

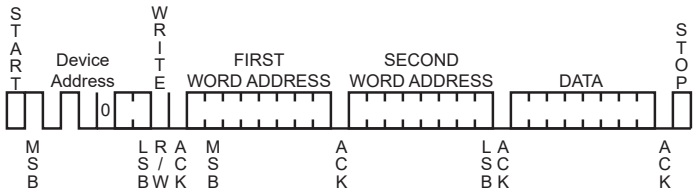
For a slave address higher than seven bits, configure the address size (IADRSZ) and set the other slave address bits in the Internal Address register (TWIHS_IADR). The two remaining internal address bytes, IADR[15:8] and IADR[23:16], can be used the same way as in 7-bit slave addressing.

Example: Address a 10-bit device (10-bit device address is b1 b2 b3 b4 b5 b6 b7 b8 b9 b10)

1. Program IADRSZ = 1,
2. Program DADR with 1 1 1 1 0 b1 b2 (b1 is the MSB of the 10-bit address, b2, etc.)
3. Program TWIHS_IADR with b3 b4 b5 b6 b7 b8 b9 b10 (b10 is the LSB of the 10-bit address)

The figure below shows a byte write to a memory device. This demonstrates the use of internal addresses to access the device.

**Figure 45-12. Internal Address Usage**




### 45.6.3.7 Repeated Start

In addition to Internal Address mode, REPEATED START (Sr) can be generated manually by writing the START bit at the end of a transfer instead of the STOP bit. In such case, the parameters of the next transfer (direction, SADR, etc.) need to be set before writing the START bit at the end of the previous transfer.

See Read/Write Flowcharts for detailed flowcharts.

Note that generating a REPEATED START after a single data read is not supported.

#### 45.6.3.8 Bus Clear Command

The TWIHS can perform a Bus Clear command:

1. Configure the Master mode (DADR, CKDIV, etc).
2. Start the transfer by setting TWIHS_CR.CLEAR.

**Note:** If alternative command is used (ACMEN bit set to '1'), DATAL field must be set to 0.

#### 45.6.3.9 Using the DMA Controller (DMAC) in Master Mode

The use of the DMA significantly reduces the CPU load.

To ensure correct implementation, follow the programming sequences below:

##### 45.6.3.9.1 Data Transmit with the DMA in Master Mode

If Alternative Command mode is disabled (ACMEN bit set to '0'):

The DMA transfer size must be defined with the buffer size minus 1. The remaining character must be managed without DMA to ensure that the exact number of bytes are transmitted regardless of system bus latency conditions during the end of the buffer transfer period.

1. Initialize the DMA (channels, memory pointers, size - 1, etc.);
2. Configure the Master mode (DADR, CKDIV, MREAD = 0, etc.) or Slave mode.
3. Enable the DMA.
4. Wait for the DMA status flag indicating that the buffer transfer is complete.
5. Disable the DMA.
6. Wait for the TXRDY flag in TWIHS_SR.
7. Set TWIHS_CR.STOP.
8. Write the last character in TWIHS_THR.
9. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWIHS_SR.

If Alternative Command mode is enabled (ACMEN bit set to '1'):

1. Initialize the transmit DMA (memory pointers, transfer size).
2. Configure the Master mode (DADR, CKDIV, etc.) and TWIHS_ACR.
3. Start the transfer by setting the DMA TXTEN bit.
4. Wait for the DMA ENDTX flag either by using the polling method or ENDTX interrupt.
5. Disable the DMA by setting the DMA TXTDIS bit.
6. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWIHS_SR.

##### 45.6.3.9.2 Data Receive with the DMA in Master Mode

If Alternative Command mode is disabled (ACMEN bit set to '0'):

The DMA transfer size must be defined with the buffer size minus 2. The two remaining characters must be managed without DMA to ensure that the exact number of bytes are received regardless of system bus latency conditions encountered during the end of buffer transfer period.

1. Initialize the DMA (channels, memory pointers, size - 2, etc.);
2. Configure the Master mode (DADR, CKDIV, MREAD = 1, etc.) or Slave mode.
3. Enable the DMA.
4. (Master Only) Write TWIHS_CR.START to start the transfer.
5. Wait for the DMA status flag indicating that the buffer transfer is complete.
6. Disable the DMA.
7. Wait for the RXRDY flag in the TWIHS_SR.

8. Set TWIHS_CR.STOP.
9. Read the penultimate character in TWIHS_RHR.
10. Wait for the RXRDY flag in the TWIHS_SR.
11. Read the last character in TWIHS_RHR.
12. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWIHS_SR.

If Alternative Command mode is enabled (ACMEN bit set to '1'):

1. Initialize the transmit DMA (memory pointers, transfer size).
2. Configure the Master mode (DADR, CKDIV, etc.) and TWIHS_ACR.
3. Set the DMA RXTEN bit.
4. (Master Only) Write TWIHS_CR.START to start the transfer.
5. Wait for the DMA ENDTX Flag either by using the polling method or ENDTX interrupt.
6. Disable the DMA by setting the DMA TXTDIS bit.
7. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWIHS_SR.

#### 45.6.3.10 SMBus Mode

SMBus mode is enabled when a one is written to TWIHS_CR.SMBEN. SMBus mode operation is similar to I²C operation with the following exceptions:

- Only 7-bit addressing can be used.
- The SMBus standard describes a set of timeout values to ensure progress and throughput on the bus. These timeout values must be programmed into TWIHS_SMBTR.
- Transmissions can optionally include a CRC byte, called Packet Error Check (PEC).
- A set of addresses has been reserved for protocol handling, such as alert response address (ARA) and host header (HH) address. Address matching on these addresses can be enabled by configuring TWIHS_CR.

##### 45.6.3.10.1 Packet Error Checking

Each SMBus transfer can optionally end with a CRC byte, called the PEC byte. Writing a one to TWIHS_CR.PECEN enables automatic PEC handling in the current transfer. Transfers with and without PEC can be intermixed in the same system, since some slaves may not support PEC. The PEC LFSR is always updated on every bit transmitted or received, so that PEC handling on combined transfers is correct.

In Master Transmitter mode, the master calculates a PEC value and transmits it to the slave after all data bytes have been transmitted. Upon reception of this PEC byte, the slave compares it to the PEC value it has computed itself. If the values match, the data was received correctly, and the slave returns an ACK to the master. If the PEC values differ, data was corrupted, and the slave returns a NACK value. Some slaves may not be able to check the received PEC in time to return a NACK if an error occurred. In this case, the slave should always return an ACK after the PEC byte, and another method must be used to verify that the transmission was received correctly.

In Master Receiver mode, the slave calculates a PEC value and transmits it to the master after all data bytes have been transmitted. Upon reception of this PEC byte, the master compares it to the PEC value it has computed itself. If the values match, the data was received correctly. If the PEC values differ, data was corrupted, and TWIHS_SR.PECERR is set. In Master Receiver mode, the PEC byte is always followed by a NACK transmitted by the master, since it is the last byte in the transfer.

In combined transfers, the PECRQ bit should only be set in the last of the combined transfers.

If the Alternative Command mode is enabled, only the NPEC bit should be set.

Consider the following transfer:

S, ADR+W, COMMAND_BYTE, ACK, SR, ADR+R, DATA_BYTE, ACK, PEC_BYTE, NACK, P

See Read/Write Flowcharts for detailed flowcharts.

#### 45.6.3.10.2 Timeouts

The TLOWS and TLOWM fields in TWIHS_SMBTR configure the SMBus timeout values. If a timeout occurs, the master transmits a STOP condition and leaves the bus. The TOUT bit is also set in TWIHS_SR.

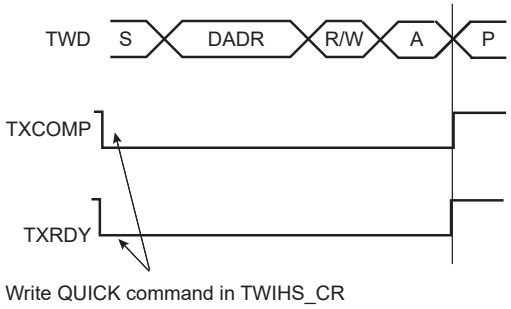
### 45.6.3.11 SMBus Quick Command (Master Mode Only)

The TWIHS can perform a quick command by following these steps:

1. Configure the Master mode (DADR, CKDIV, etc).
2. Write TWIHS_MMR.MREAD at the value of the one-bit command to be sent.
3. Start the transfer by setting TWIHS_CR.QUICK.

If alternative command is used (ACMEN bit set to ‘1’), DATAL field must be set to 0.

**Figure 45-13. SMBus Quick Command**




### 45.6.3.12 Alternative Command

Another way to configure the transfer is to enable the Alternative Command mode with the ACMEN bit of the TWIHS Control Register.

In this mode, the transfer is configured through the TWIHS Alternative Command Register. It is possible to define a simple read or write transfer or a combined transfer with a repeated start.

In order to set a simple transfer, the DATAL field and the DIR field of the TWIHS Alternative Command Register must be filled accordingly and the NDATAL field must be cleared. To begin the transfer, either set the START bit in the TWIHS Control Register in case of a read transfer, or write the TWIHS Transmit Holding Register in case of a write transfer.

For a combined transfer linked by a repeated start, the NDATAL field must be filled with the length of the second transfer and NDIR with the corresponding direction.

The PEC and NPEC bits are used to set a PEC field. In the case of a single transfer with PEC, the PEC bit must be set. In the case of a combined transfer, the NPEC bit must be set.

**Note:** If Alternative Command mode is used, TWIHS_MMR.IADRSZ must be set to 0.

See Read/Write Flowcharts for detailed flowcharts.

### 45.6.3.13 Handling Errors in Alternative Command

If a NACK is generated by a slave device or SMBus timeout error, the TWIHS stops the frame immediately, although the DMA transfer may still be active. To prevent a new frame from being restarted

with the remaining DMA data (transmit), the TWIHS prevents any start of frame until the LOCK flag is cleared in the TWIHS_SR.

TWIHS_SR.LOCK indicates the state of the TWIHS (locked or not locked).

When the TWIHS is locked, no transfer begins until the LOCK is cleared by TWIHS_CR.LOCKCLR and error flags are cleared by reading the TWIHS_SR.

In case of error, the TWIHS_THR may have been loaded with a new data. TWIHS_CR.THRCLR can be used to flush the TWIHS_THR. If THRCLR is set, the TXRDY and TXCOMP flags are set.

#### 45.6.3.14 Read/Write Flowcharts

The flowcharts give examples for read and write operations. A polling or interrupt method can be used to check the status bits. The interrupt method requires that TWIHS_IER be configured first.

**Figure 45-14. TWIHS Write Operation with Single Data Byte without Internal Address**

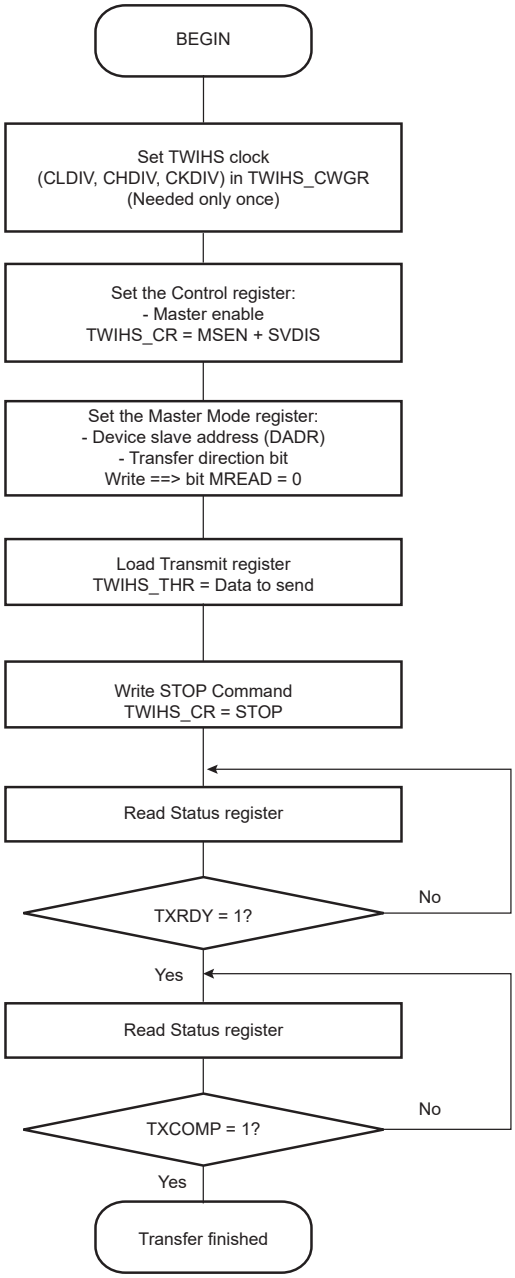


**Figure 45-15. TWIHS Write Operation with Single Data Byte and Internal Address**

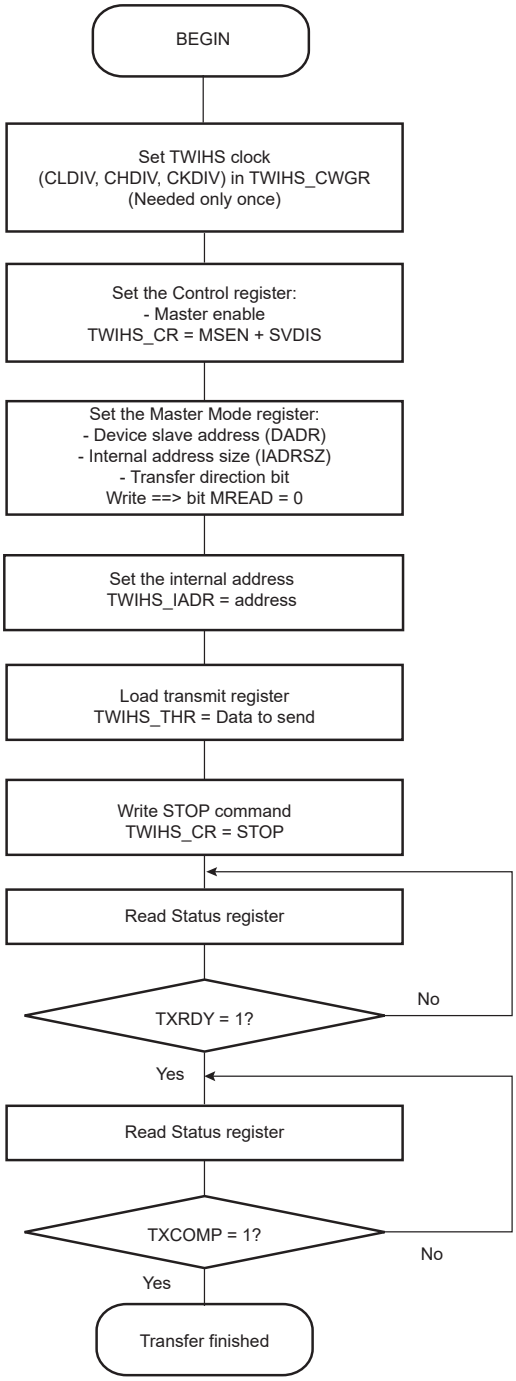


**Figure 45-16. TWIHS Write Operation with Multiple Data Bytes with or without Internal Address**

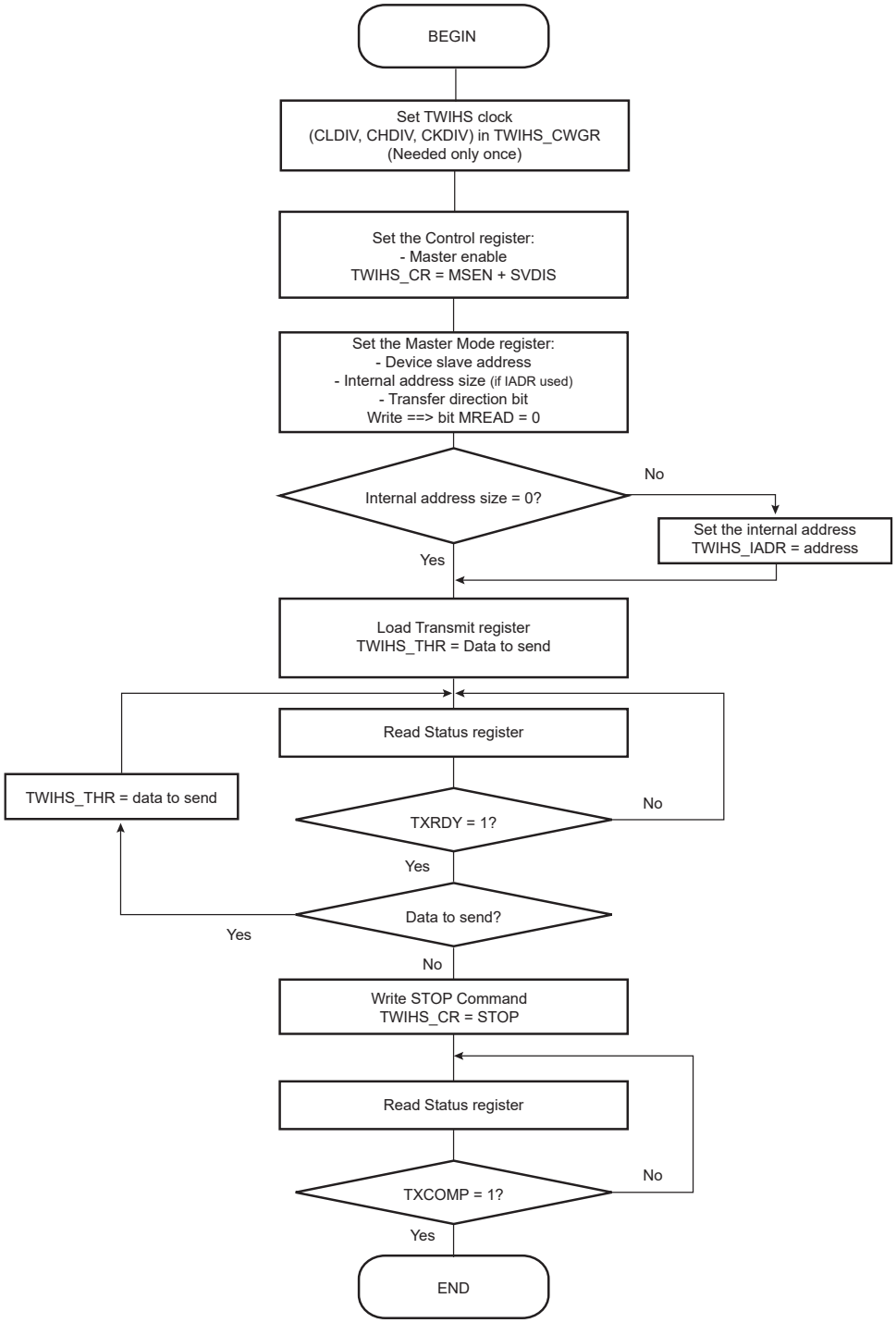


**Figure 45-17. SMBus Write Operation with Multiple Data Bytes with or without Internal Address and PEC Sending**

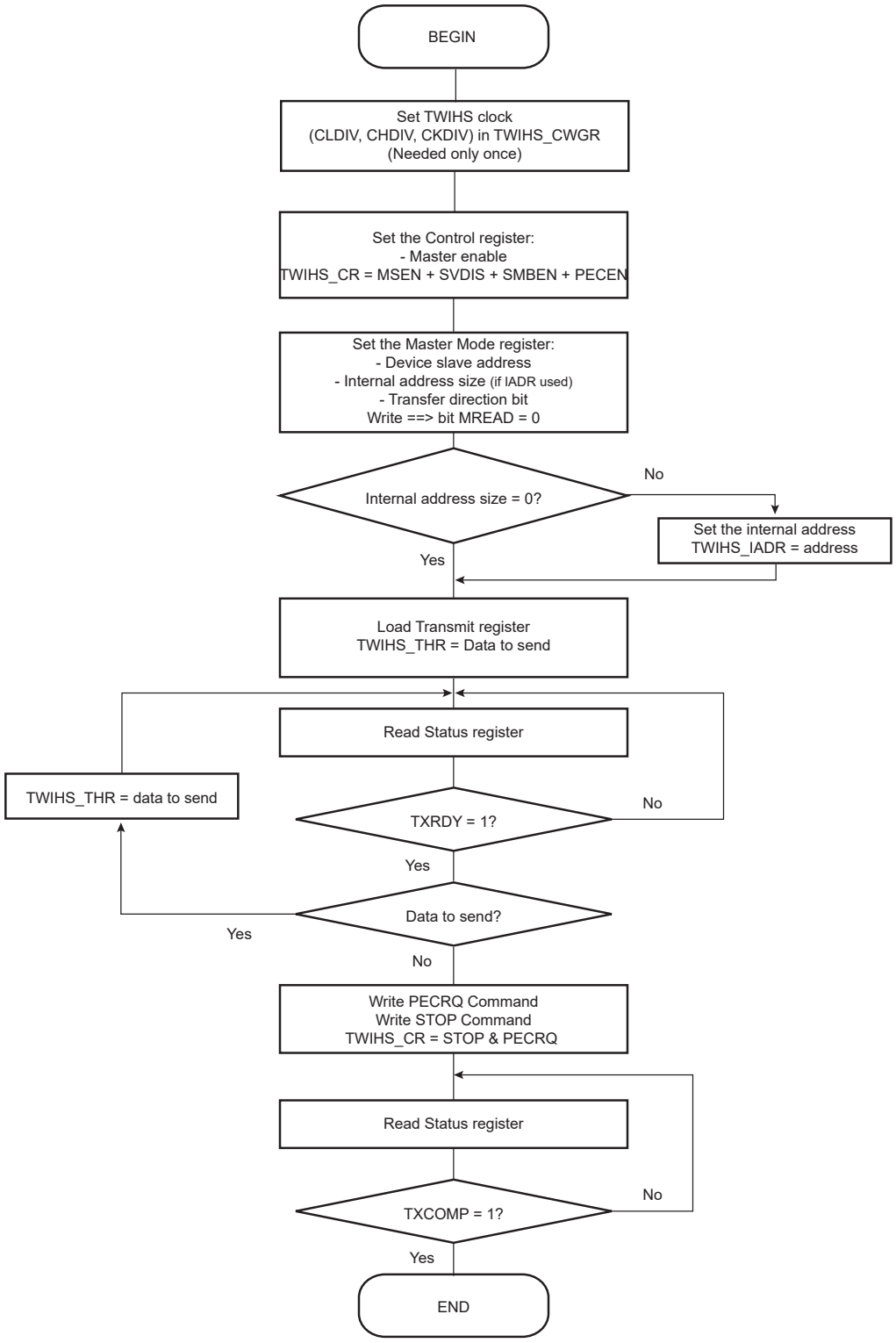


**Figure 45-18. SMBus Write Operation with Multiple Data Bytes with PEC and Alternative Command Mode**

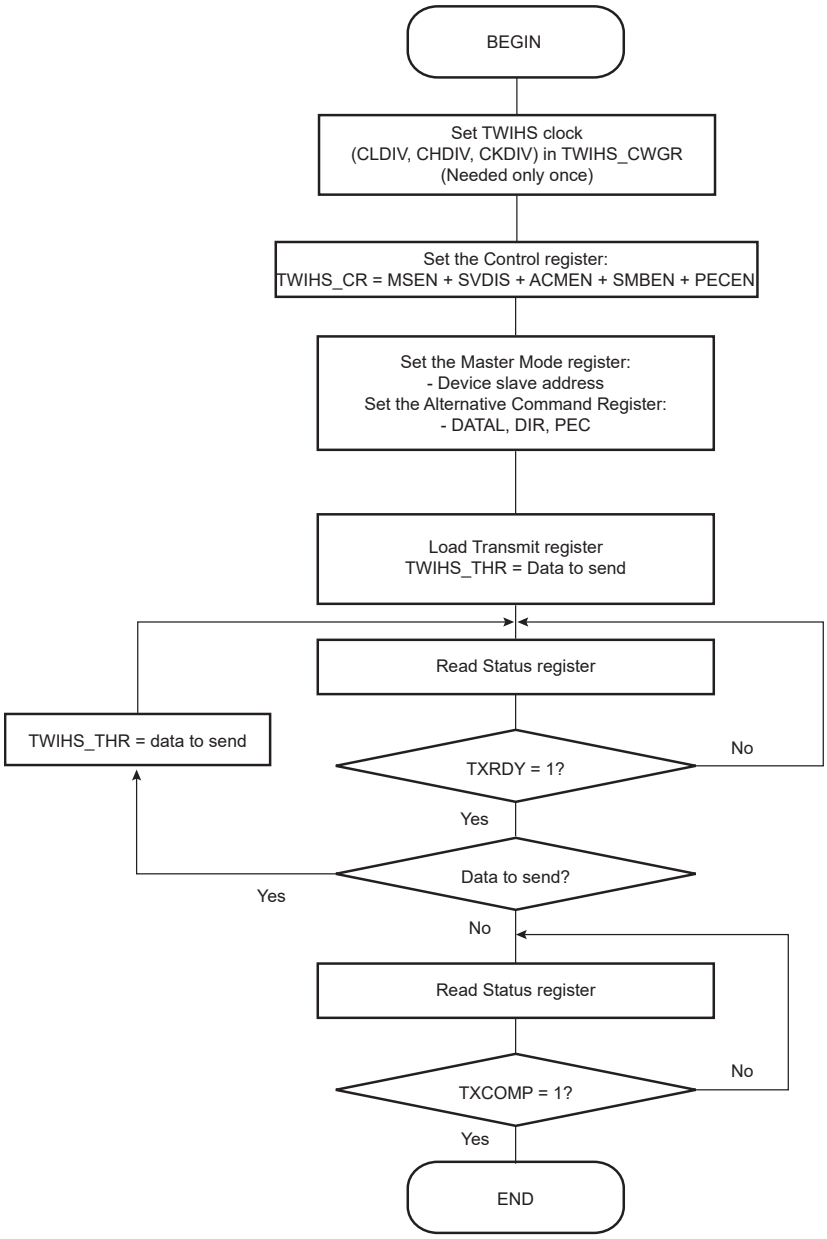


**Figure 45-19. TWIHS Write Operation with Multiple Data Bytes and Read Operation with Multiple Data Bytes (Sr)**

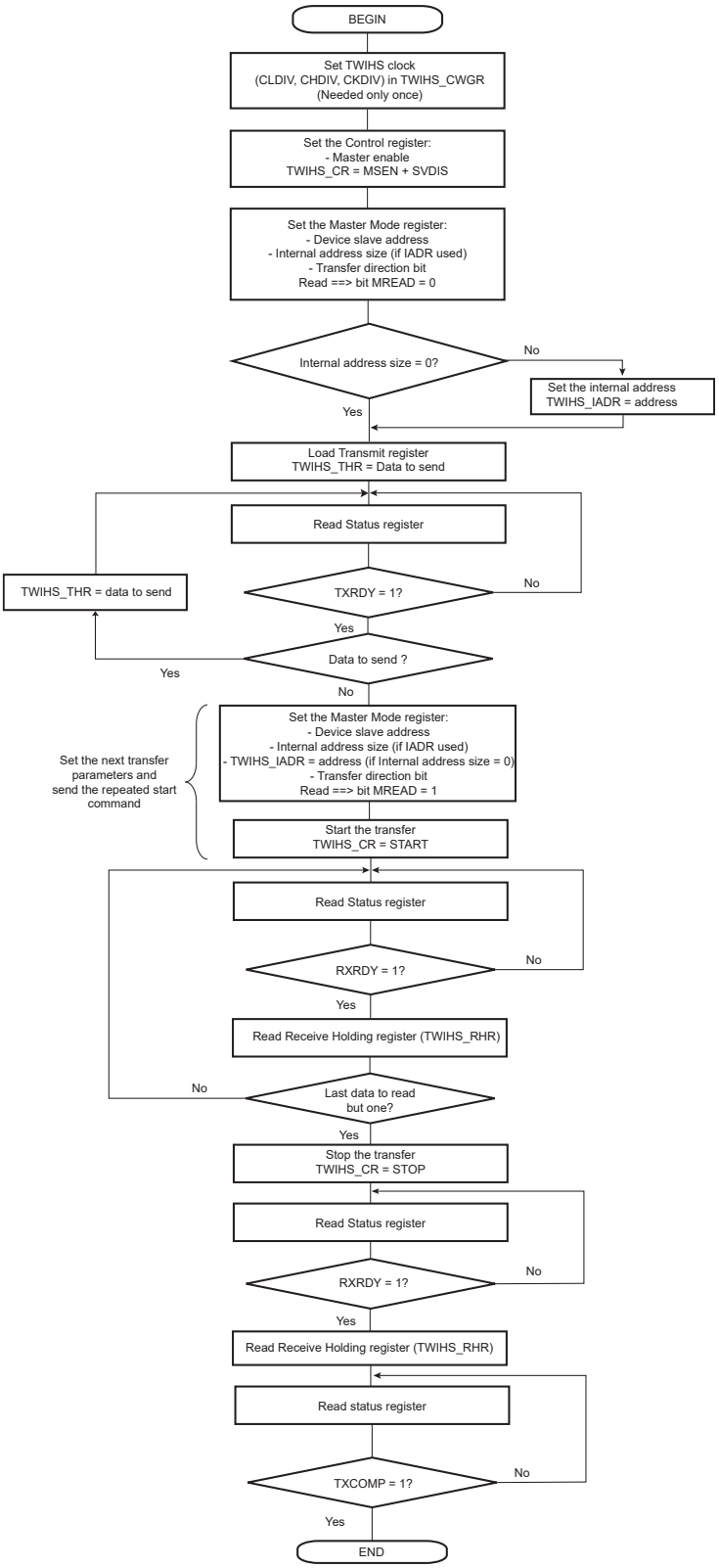


**Figure 45-20. TWIHS Write Operation with Multiple Data Bytes + Read Operation and Alternative Command Mode + PEC**

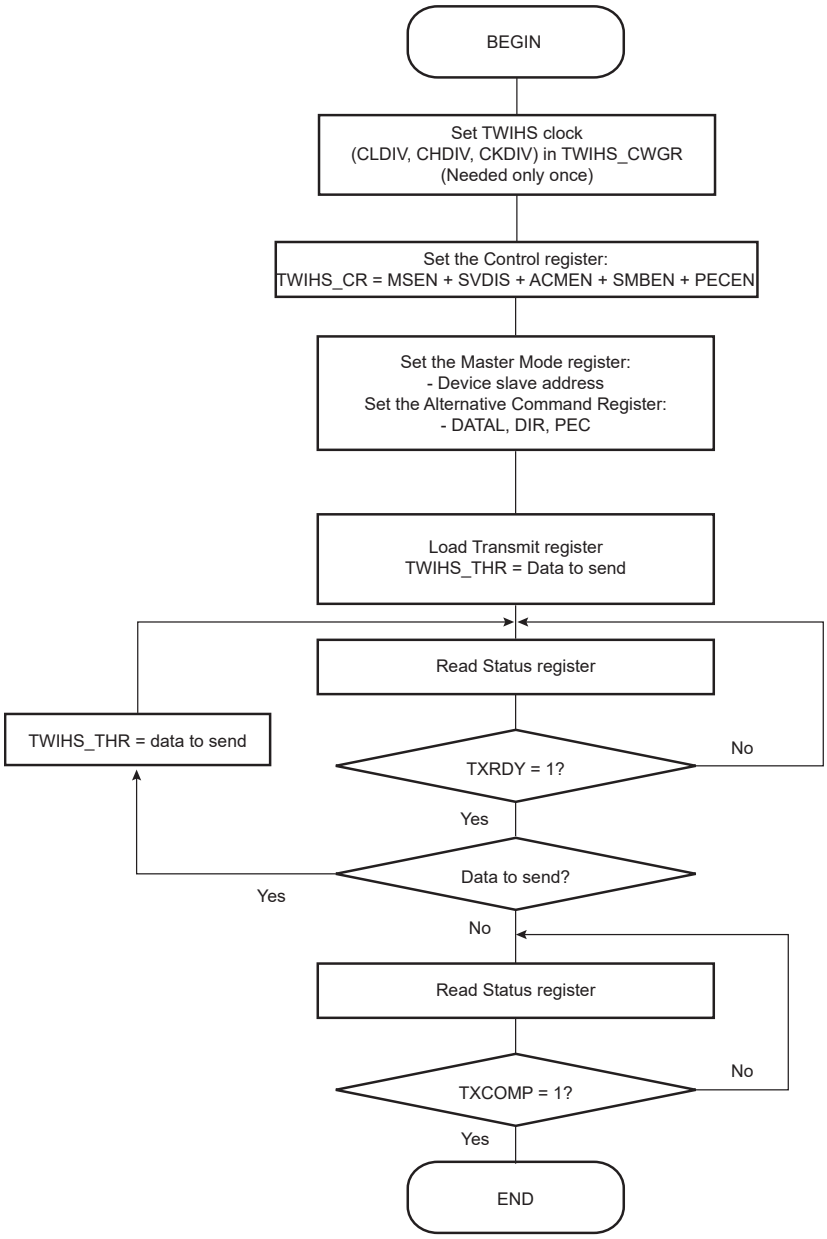


**Figure 45-21. TWIHS Read Operation with Single Data Byte without Internal Address**

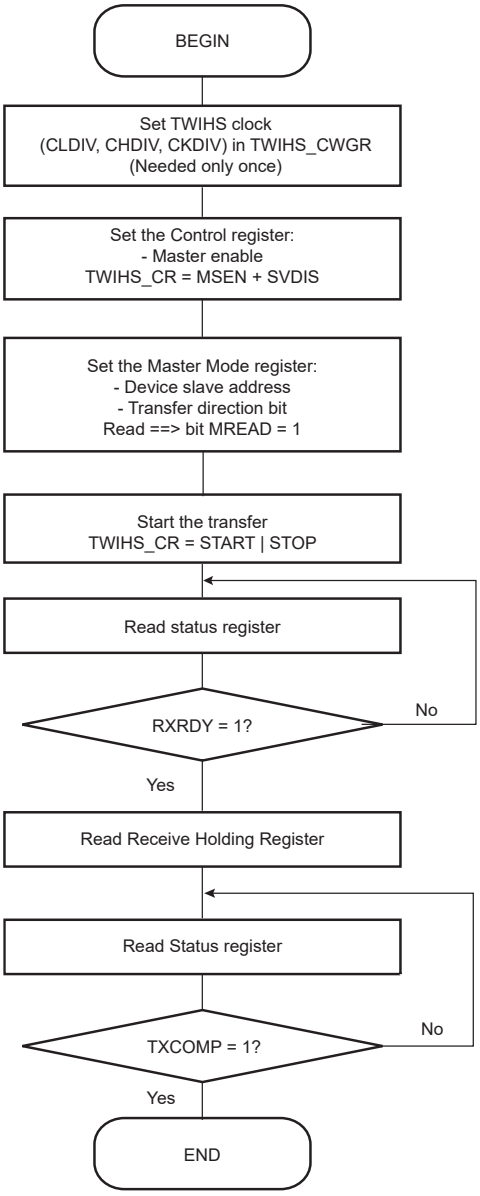


**Figure 45-22. TWIHS Read Operation with Single Data Byte and Internal Address**

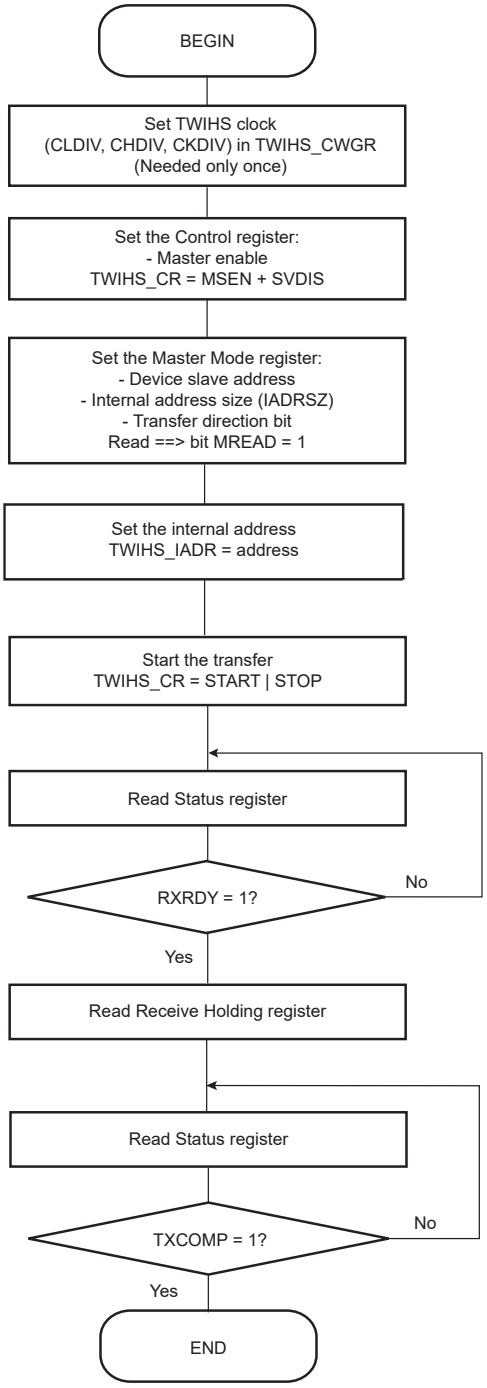


**Figure 45-23. TWIHS Read Operation with Multiple Data Bytes with or without Internal Address**

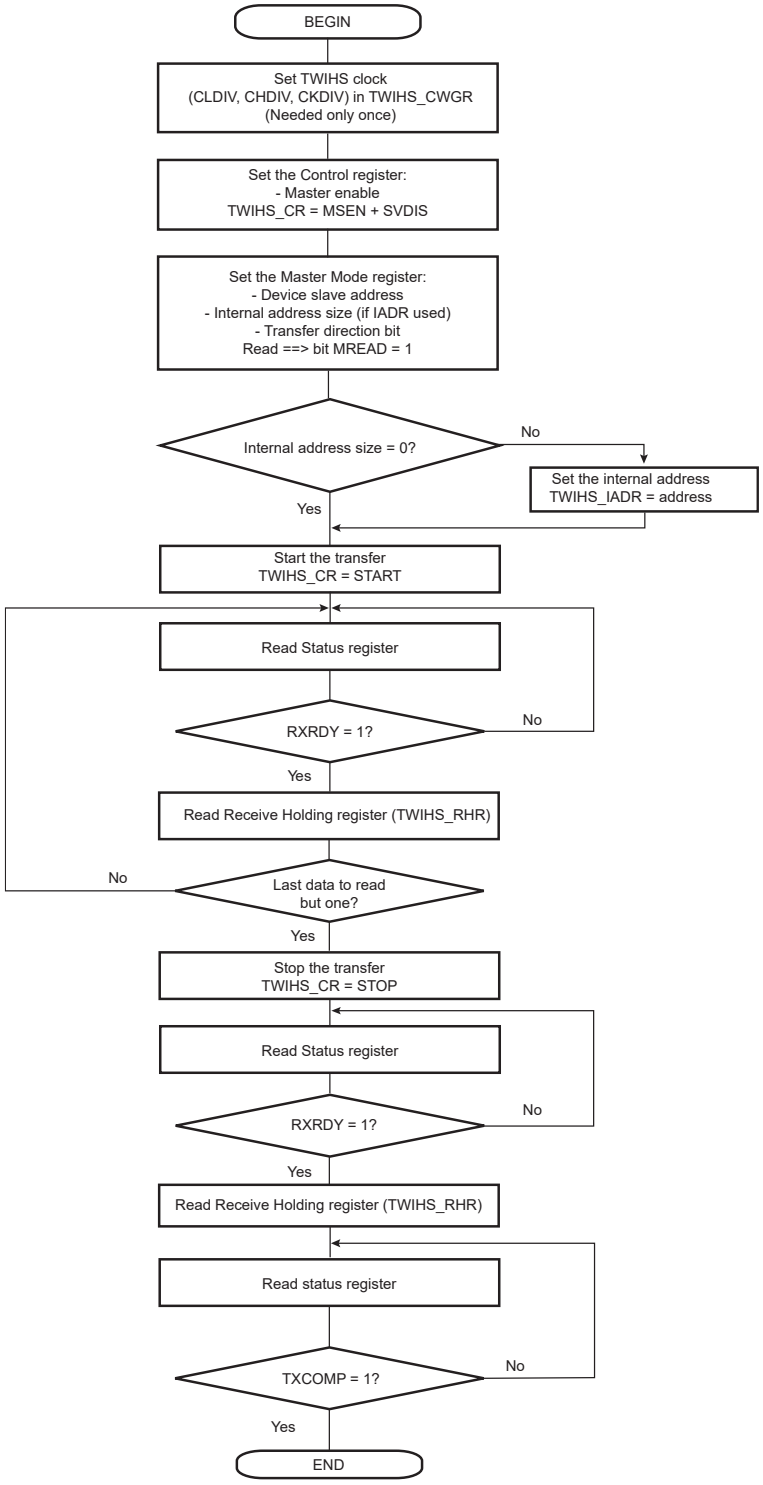


**Figure 45-24. TWIHS Read Operation with Multiple Data Bytes with or without Internal Address with PEC**

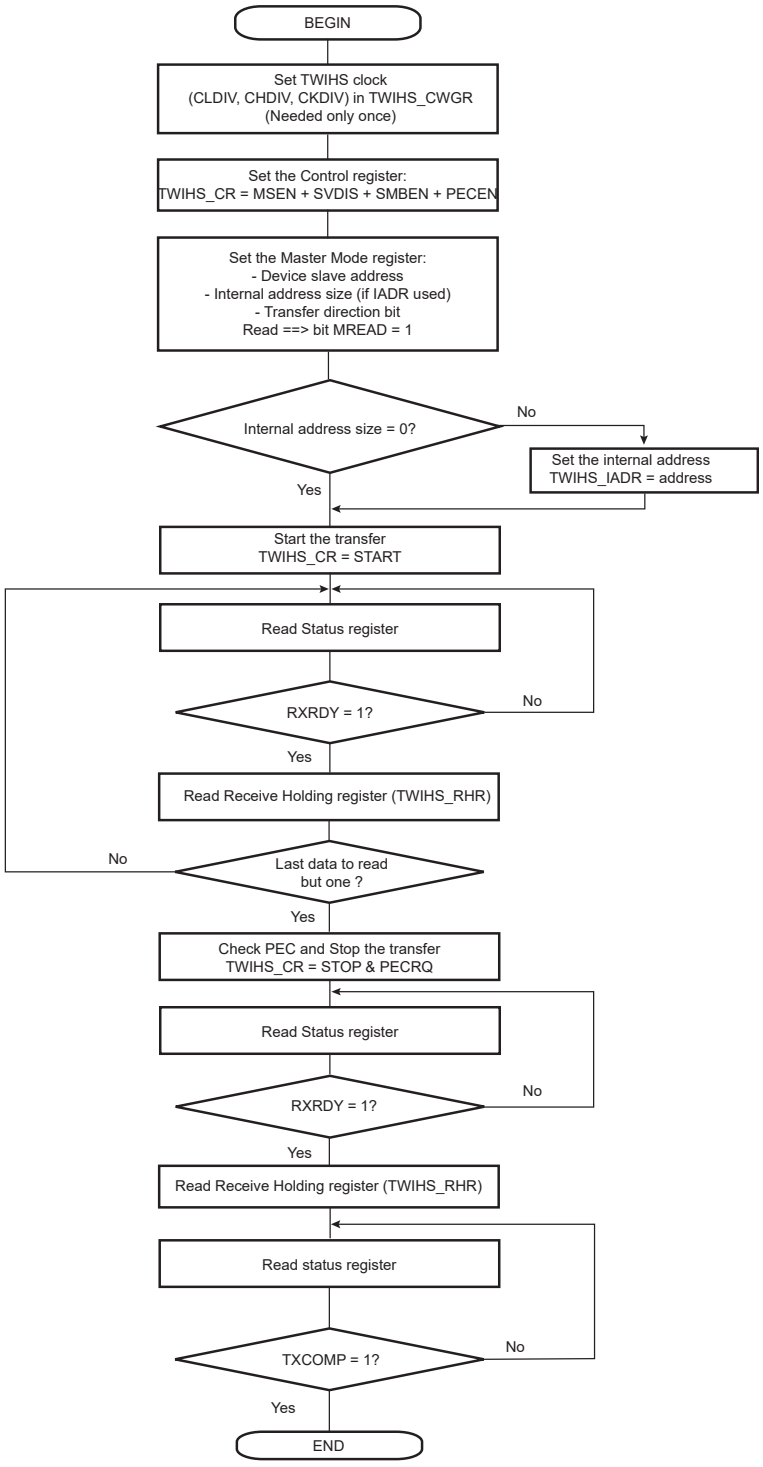


**Figure 45-25. TWIHS Read Operation with Multiple Data Bytes with Alternative Command Mode with PEC**

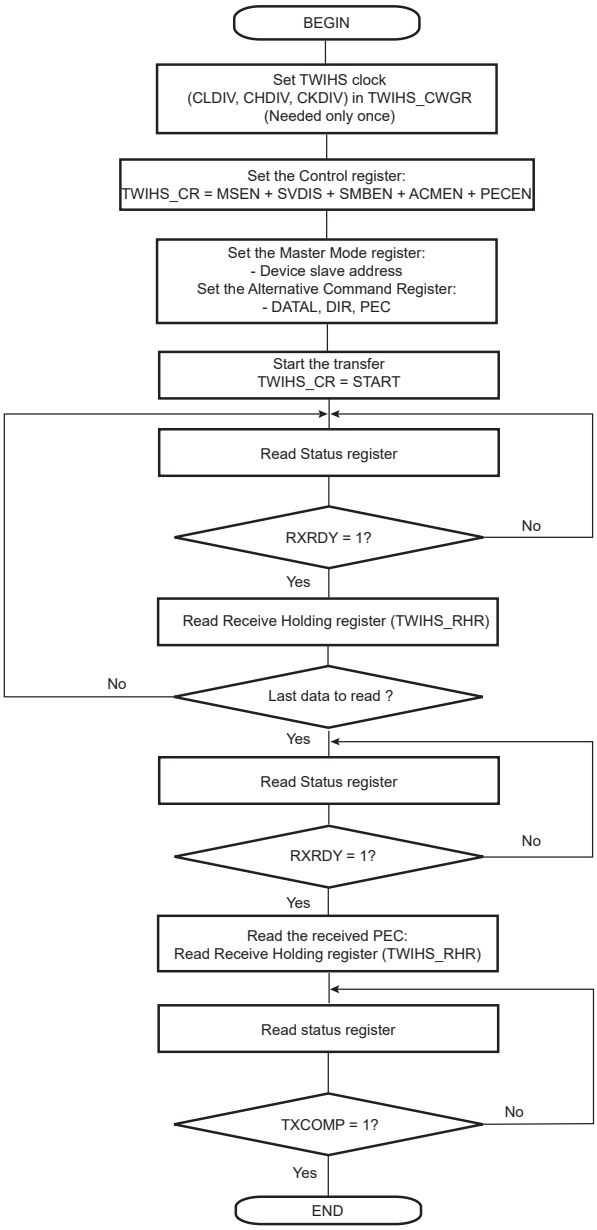


**Figure 45-26. TWIHS Read Operation with Multiple Data Bytes + Write Operation with Multiple Data Bytes (Sr)**

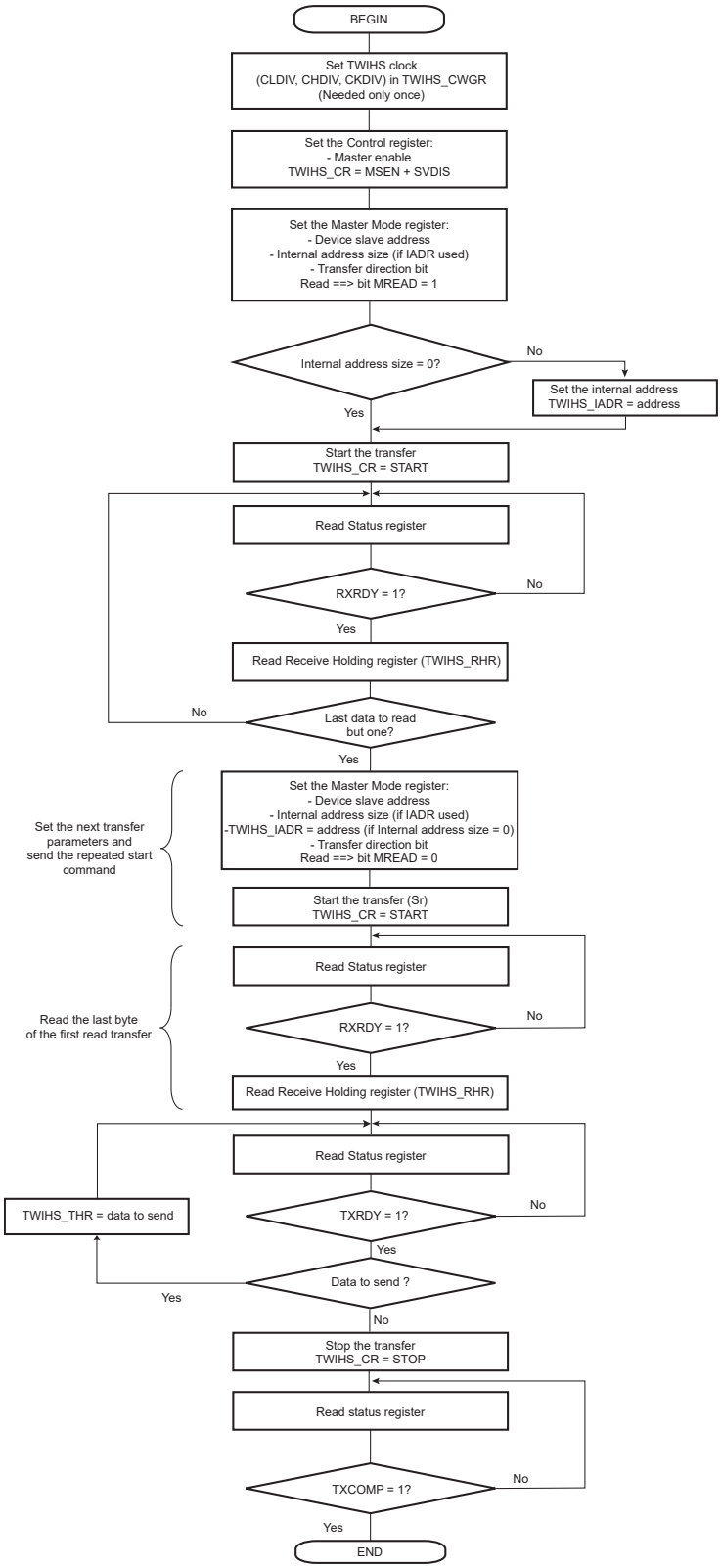
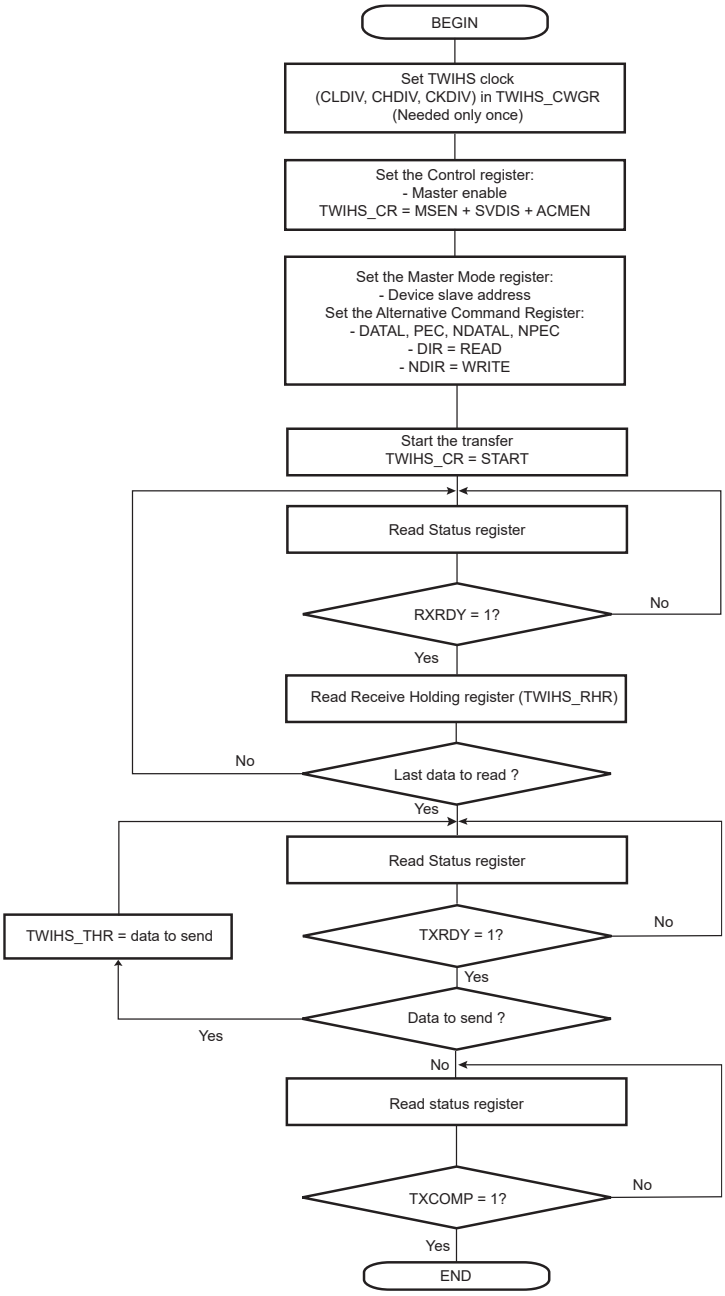


**Figure 45-27. TWIHS Read Operation with Multiple Data Bytes + Write with Alternative Command Mode with PEC**




### 45.6.4 Multimaster Mode

#### 45.6.4.1 Definition

In Multimaster mode, more than one master may handle the bus at the same time without data corruption by using arbitration.

Arbitration starts as soon as two or more masters place information on the bus at the same time, and stops (arbitration is lost) for the master that intends to send a logical one while the other master sends a logical zero.

As soon as arbitration is lost by a master, it stops sending data and listens to the bus in order to detect a stop. When the stop is detected, the master that has lost arbitration may put its data on the bus by respecting arbitration.

Arbitration is illustrated in Arbitration Cases.

#### 45.6.4.2 Different Multimaster Modes

Two Multimaster modes may be distinguished:

1. The TWIHS is considered as a master only and is never addressed.
2. The TWIHS may be either a master or a slave and may be addressed.

**Note:** Arbitration in supported in both Multimaster modes.

##### 45.6.4.2.1 TWIHS as Master Only

In this mode, the TWIHS is considered as a master only (MSEN is always at one) and must be driven like a master with the ARBLST (Arbitration Lost) flag in addition.

If arbitration is lost (ARBLST = 1), the user must reinitiate the data transfer.

If starting a transfer (ex.: DADR + START + W + Write in THR) and if the bus is busy, the TWIHS automatically waits for a STOP condition on the bus to initiate the transfer (see User Sends Data While the Bus is Busy).

**Note:** The state of the bus (busy or free) is not indicated in the user interface.

##### 45.6.4.2.2 TWIHS as Master or Slave

The automatic reversal from master to slave is not supported in case of a lost arbitration.

Then, in the case where TWIHS may be either a master or a slave, the user must manage the pseudo Multimaster mode described in the steps below:

1. Program the TWIHS in Slave mode (SADR + MSDIS + SVEN) and perform a slave access (if TWIHS is addressed).
2. If the TWIHS has to be set in Master mode, wait until TXCOMP flag is at 1.
3. Program the Master mode (DADR + SVDIS + MSEN) and start the transfer (ex: START + Write in THR).
4. As soon as the Master mode is enabled, the TWIHS scans the bus in order to detect if it is busy or free. When the bus is considered free, the TWIHS initiates the transfer.
5. As soon as the transfer is initiated and until a STOP condition is sent, the arbitration becomes relevant and the user must monitor the ARBLST flag.
6. If the arbitration is lost (ARBLST is set to 1), the user must program the TWIHS in Slave mode in case the master that won the arbitration needs to access the TWIHS.
7. If the TWIHS has to be set in Slave mode, wait until the TXCOMP flag is at 1 and then program the Slave mode.

**Note:** If the arbitration is lost and the TWIHS is addressed, the TWIHS does not acknowledge, even if it is programmed in Slave mode as soon as ARBLST is set to 1. Then the master must repeat SADR.

**Figure 45-28. User Sends Data While the Bus is Busy**

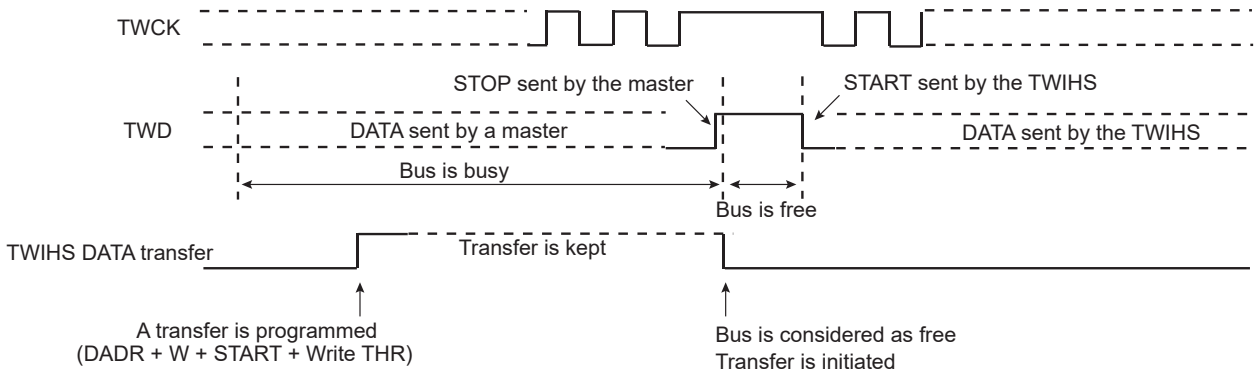
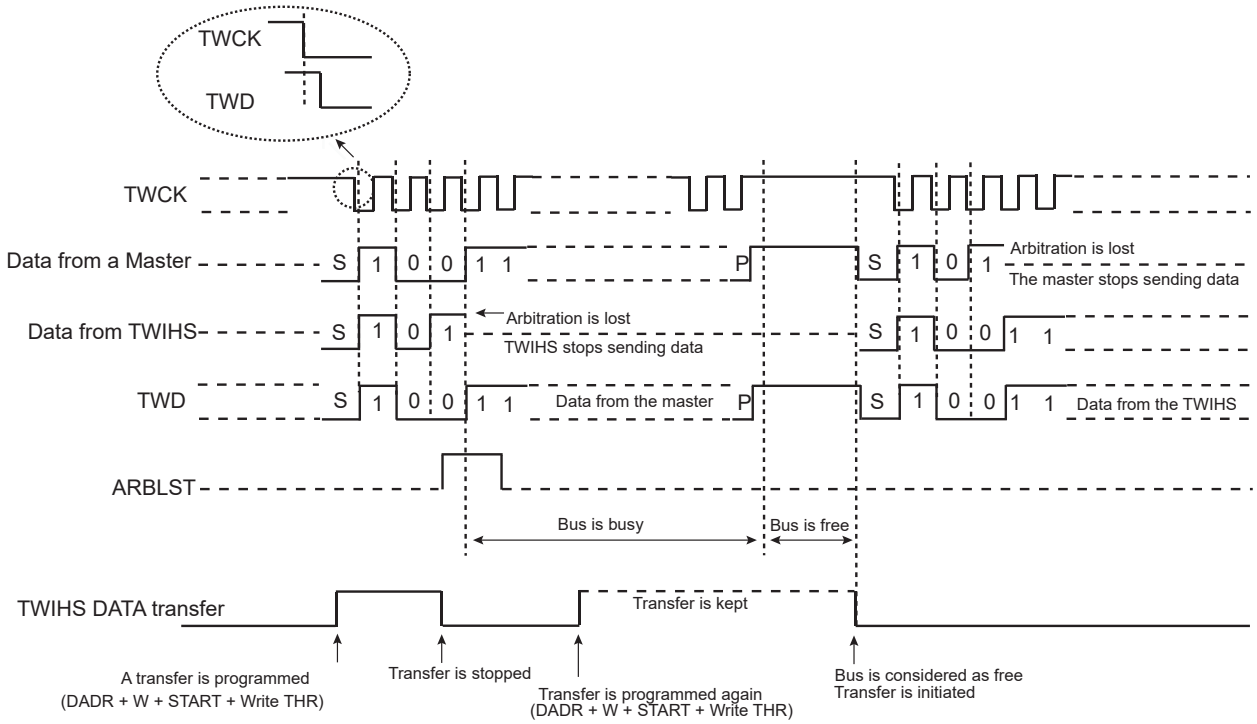
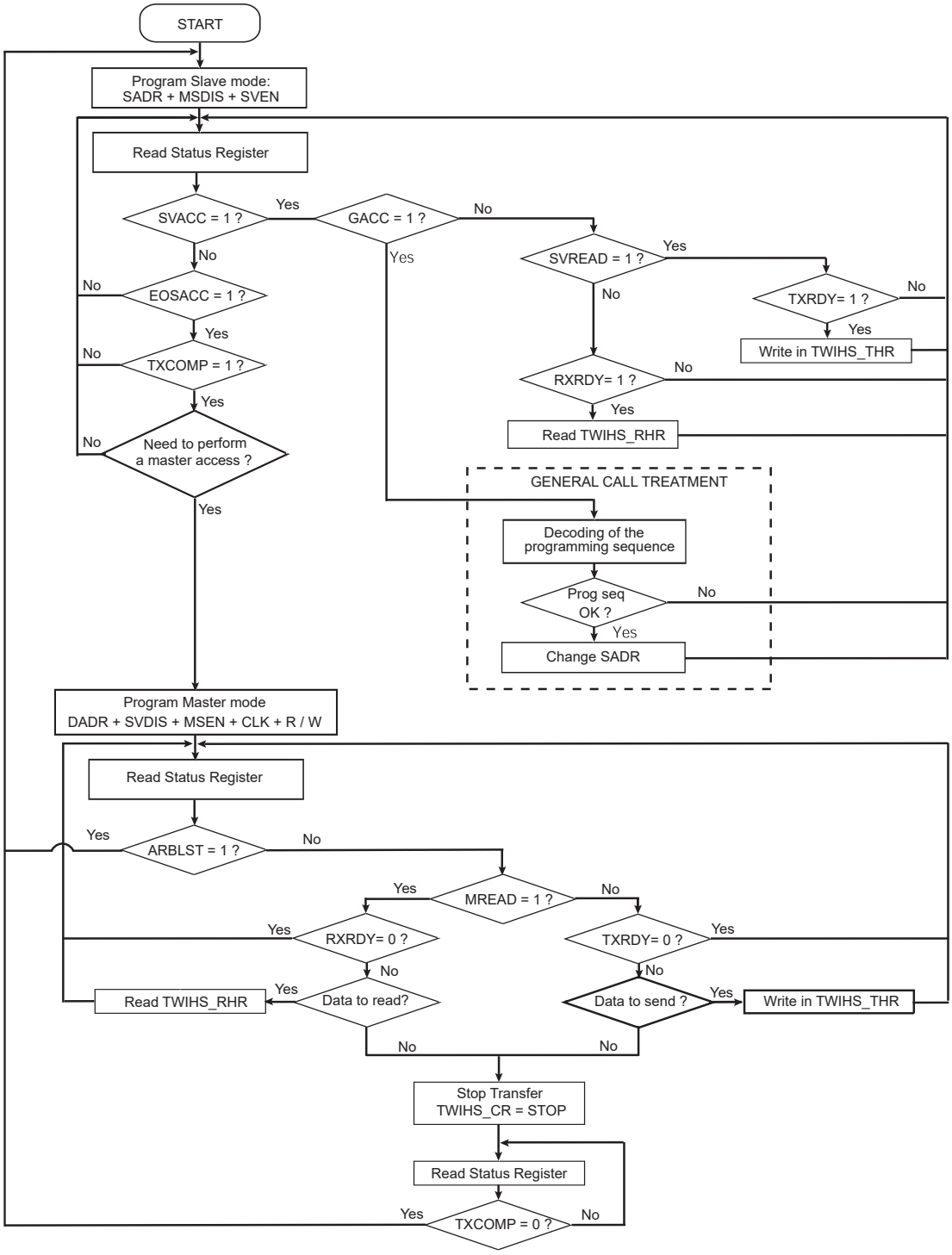



**Figure 45-29. Arbitration Cases**




The flowchart below gives an example of read and write operations in Multimaster mode.

**Figure 45-30. Multimaster Flowchart**




## 45.6.5 Slave Mode

### 45.6.5.1 Definition

Slave mode is defined as a mode where the device receives the clock and the address from another device called the master.

In this mode, the device never initiates and never completes the transmission (START, REPEATED_START and STOP conditions are always provided by the master).

#### 45.6.5.2 Programming Slave Mode

The following fields must be programmed before entering Slave mode:

1. TWIHS_SMR.SADR: The slave device address is used in order to be accessed by master devices in Read or Write mode.
2. (Optional) TWIHS_SMR.MASK can be set to mask some SADR address bits and thus allow multiple address matching.
3. TWIHS_CR.MSDIS: Disables the Master mode.
4. TWIHS_CR.SVEN: Enables the Slave mode.

As the device receives the clock, values written in TWIHS_CWGR are ignored.

#### 45.6.5.3 Receiving Data

After a START or REPEATED START condition is detected, and if the address sent by the master matches the slave address programmed in the SADR (Slave Address) field, the SVACC (Slave Access) flag is set and SVREAD (Slave Read) indicates the direction of the transfer.

SVACC remains high until a STOP condition or a REPEATED START is detected. When such a condition is detected, the EOSACC (End Of Slave Access) flag is set.

##### 45.6.5.3.1 Read Sequence

In the case of a read sequence (SVREAD is high), the TWIHS transfers data written in the TWIHS_THR until a STOP condition or a REPEATED_START + an address different from SADR is detected. Note that at the end of the read sequence TXCOMP (Transmission Complete) flag is set and SVACC reset.

As soon as data is written in the TWIHS_THR, TXRDY (Transmit Holding Register Ready) flag is reset, and it is set when the internal shifter is empty and the sent data acknowledged or not. If the data is not acknowledged, the NACK flag is set.

Note that a STOP or a REPEATED START always follows a NACK.

To clear the TXRDY flag, first set TWIHS_CR.SVDIS, then set TWIHS_CR.SVEN.

See Read Access Ordered by a Master.

##### 45.6.5.3.2 Write Sequence

In the case of a write sequence (SVREAD is low), the RXRDY (Receive Holding Register Ready) flag is set as soon as a character has been received in TWIHS_RHR. RXRDY is reset when reading TWIHS_RHR.

The TWIHS continues receiving data until a STOP condition or a REPEATED_START + an address different from SADR is detected. Note that at the end of the write sequence, the TXCOMP flag is set and SVACC is reset.

See Write Access Ordered by a Master.

##### 45.6.5.3.3 Clock Stretching Sequence

If TWIHS_THR or TWIHS_RHR is not written/read in time, the TWIHS performs a clock stretching.

Clock stretching information is given by the SCLWS (Clock Wait State) bit.

See Clock Stretching in Read Mode and Clock Stretching in Write Mode.

**Note:** Clock stretching can be disabled by configuring the SCLWSDIS bit in TWIHS_SMR. In that case, the UNRE and OVRE flags indicate an underrun (when TWIHS_THR is not filled on time) or an overrun (when TWIHS_RHR is not read on time).

#### 45.6.5.3.4 General Call

In the case where a GENERAL CALL is performed, the GACC (General Call Access) flag is set.

After GACC is set, the user must interpret the meaning of the GENERAL CALL and decode the new address programming sequence.

See Master Performs a General Call.

### 45.6.5.4 Data Transfer

#### 45.6.5.4.1 Read Operation

The Read mode is defined as a data requirement from the master.

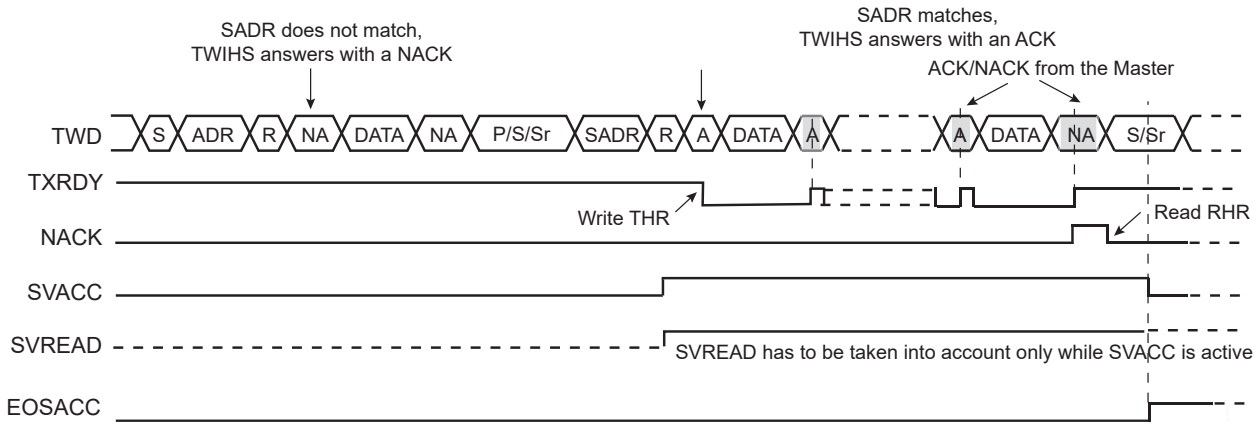
After a START or a REPEATED START condition is detected, the decoding of the address starts. If the slave address (SADR) is decoded, SVACC is set and SVREAD indicates the direction of the transfer.

Until a STOP or REPEATED START condition is detected, the TWIHS continues sending data loaded in TWIHS_THR.

If a STOP condition or a REPEATED START + an address different from SADR is detected, SVACC is reset.

The figure below describes the read operation.

**Figure 45-31. Read Access Ordered by a Master**




**Note:**

1. When SVACC is low, the state of SVREAD becomes irrelevant.
2. TXRDY is reset when data has been transmitted from TWIHS_THR to the internal shifter and set when this data has been acknowledged or non acknowledged.

#### 45.6.5.4.2 Write Operation

The Write mode is defined as a data transmission from the master.

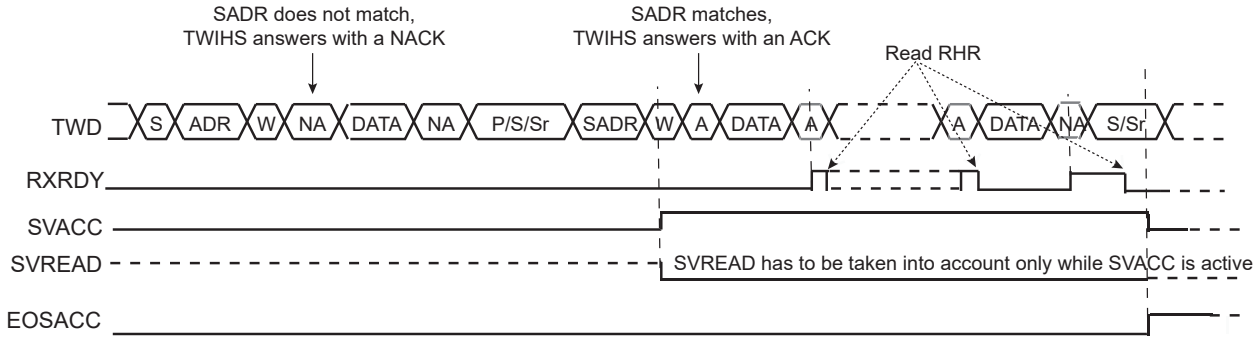
After a START or a REPEATED START, the decoding of the address starts. If the slave address is decoded, SVACC is set and SVREAD indicates the direction of the transfer (SVREAD is low in this case).

Until a STOP or REPEATED START condition is detected, the TWIHS stores the received data in TWIHS_RHR.

If a STOP condition or a REPEATED START + an address different from SADR is detected, SVACC is reset.

The figure below describes the write operation.

**Figure 45-32. Write Access Ordered by a Master**




**Note:**

1. When SVACC is low, the state of SVREAD becomes irrelevant.
2. RXRDY is set when data has been transmitted from the internal shifter to TWIHS_RHR and reset when this data is read.

#### 45.6.5.4.3 General Call

The general call is performed in order to change the address of the slave.

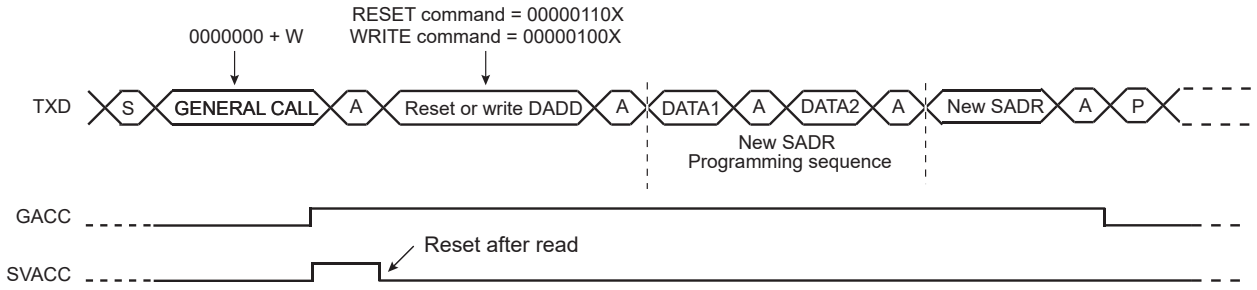
If a GENERAL CALL is detected, GACC is set.

After the detection of general call, decode the commands that follow.

In case of a WRITE command, decode the programming sequence and program a new SADR if the programming sequence matches.

The figure below describes the general call access.

**Figure 45-33. Master Performs a General Call**




**Note:** This method enables the user to create a personal programming sequence by choosing the programming bytes and their number. The programming sequence has to be provided to the master.

#### 45.6.5.4.4 Clock Stretching

In both Read and Write modes, it may occur that TWIHS_THR/TWIHS_RHR buffer is not filled/emptied before the transmission/reception of a new character. In this case, to avoid sending/receiving undesired data, a clock stretching mechanism is implemented.

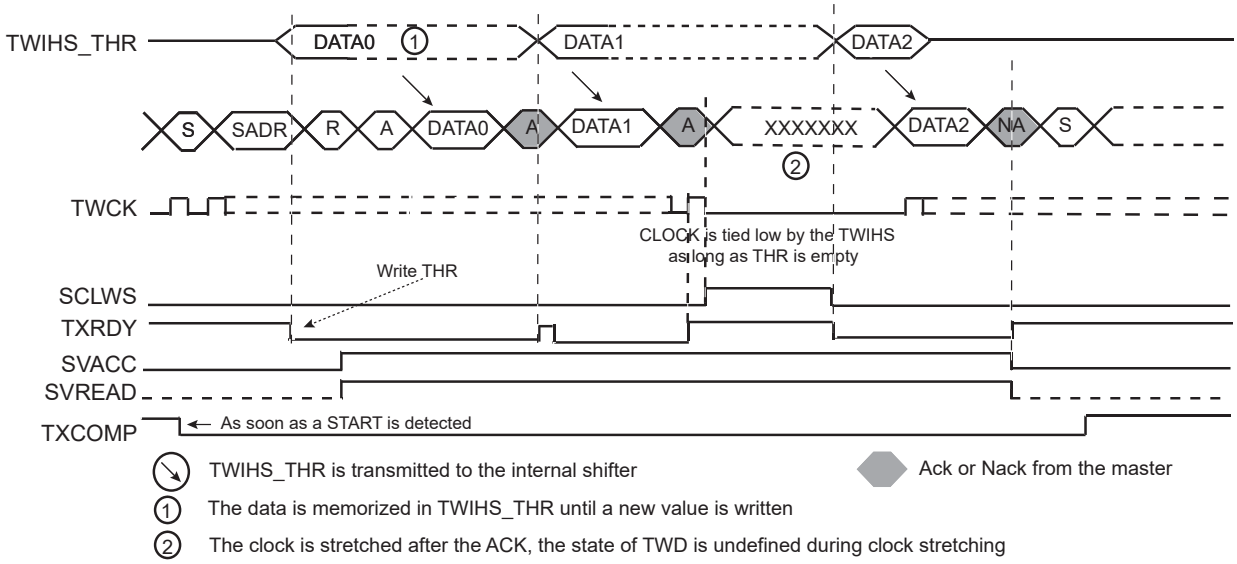
**Note:** Clock stretching can be disabled by setting TWIHS_SMR.SCLWSDIS. In that case the UNRE and OVRE flags indicate an underrun (when TWIHS_THR is not filled on time) or an overrun (when TWIHS_RHR is not read on time).

***Clock Stretching in Read Mode***

The clock is tied low if the internal shifter is empty and if a STOP or REPEATED START condition was not detected. It is tied low until the internal shifter is loaded.

The figure below describes the clock stretching in Read mode.

**Figure 45-34. Clock Stretching in Read Mode**




**Note:**

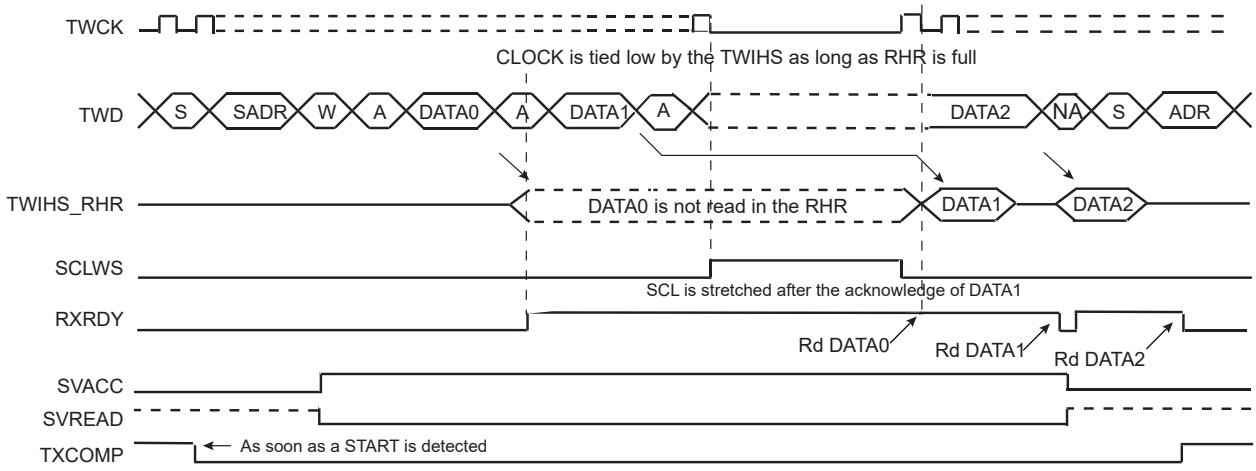
1. TXRDY is reset when data has been written in TWIHS_THR to the internal shifter and set when this data has been acknowledged or non acknowledged.
2. At the end of the read sequence, TXCOMP is set after a STOP or after a REPEATED_START + an address different from SADR.
3. SCLWS is automatically set when the clock stretching mechanism is started.

***Clock Stretching in Write Mode***

The clock is tied low if the internal shifter and TWIHS_RHR is full. If a STOP or REPEATED_START condition was not detected, it is tied low until TWIHS_RHR is read.

The figure below describes the clock stretching in Write mode.

**Figure 45-35. Clock Stretching in Write Mode**




**Note:**

1. At the end of the read sequence, TXCOMP is set after a STOP or after a REPEATED_START + an address different from SADR.
2. SCLWS is automatically set when the clock stretching mechanism is started and automatically reset when the mechanism is finished.

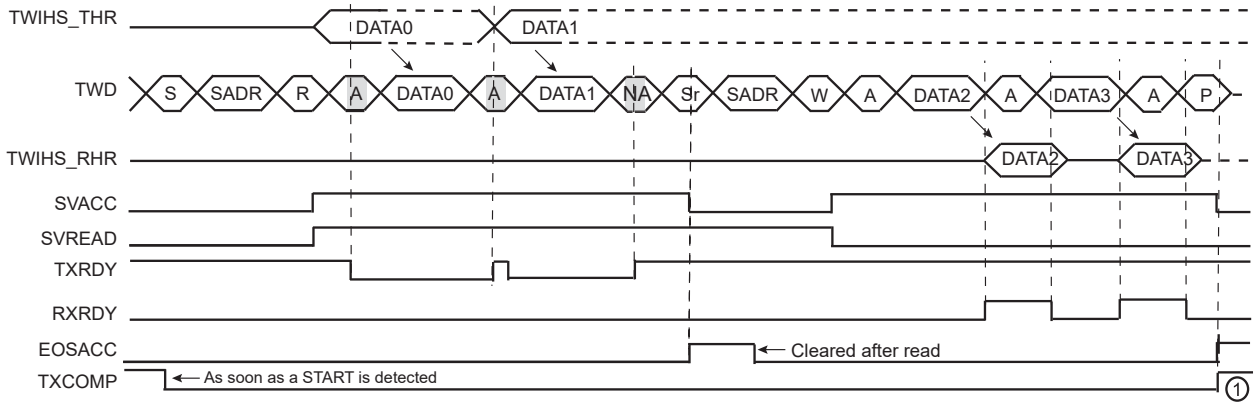
#### 45.6.5.4.5 Reversal after a Repeated Start

***Reversal of Read to Write***

The master initiates the communication by a read command and finishes it by a write command.

The figure below describes the REPEATED START and the reversal from Read mode to Write mode.

**Figure 45-36. Repeated Start and Reversal from Read Mode to Write Mode**

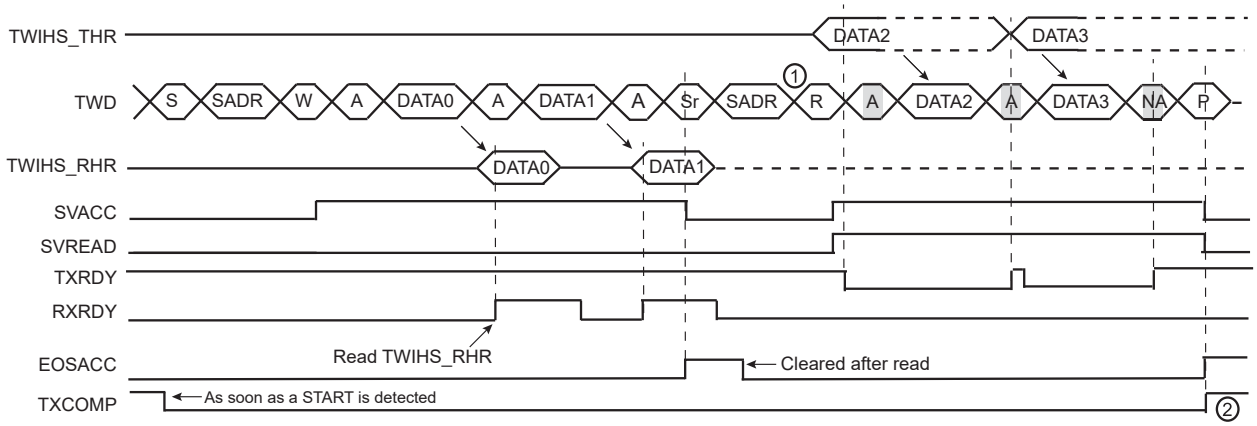



**Note:** TXCOMP is only set at the end of the transmission. This is because after the REPEATED START, SADR is detected again.

***Reversal of Write to Read***

The master initiates the communication by a write command and finishes it by a read command. The figure below describes the REPEATED START and the reversal from Write mode to Read mode.

**Figure 45-37. Repeated Start and Reversal from Write Mode to Read Mode**




**Note:**

1. In this case, if TWIHS_THR has not been written at the end of the read command, the clock is automatically stretched before the ACK.
2. TXCOMP is only set at the end of the transmission. This is because after the REPEATED START, SADR is detected again.

### 45.6.5.5 Using the DMA Controller (DMAC) in Slave Mode

The use of the DMAC significantly reduces the CPU load.

#### 45.6.5.5.1 Data Transmit with the DMA in Slave Mode

The following procedure shows an example to transmit data with DMA.

1. Initialize the transmit DMA (memory pointers, transfer size, etc).

2. Configure the Slave mode.
3. Enable the DMA.
4. Wait for the DMA status flag indicating that the buffer transfer is complete.
5. Disable the DMA.
6. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWIHS_SR.

#### 45.6.5.5.2 Data Receive with the DMA in Slave Mode

The following procedure shows an example to transmit data with DMA where the number of characters to receive is known.

1. Initialize the DMA (channels, memory pointers, size, etc.).
2. Configure the Slave mode.
3. Enable the DMA.
4. Wait for the DMA status flag indicating that the buffer transfer is complete.
5. Disable the DMA.
6. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWIHS_SR.

### 45.6.5.6 SMBus Mode

SMBus mode is enabled when a one is written to TWIHS_CR.SMBEN. SMBus mode operation is similar to I²C operation with the following exceptions:

- Only 7-bit addressing can be used.
- The SMBus standard describes a set of timeout values to ensure progress and throughput on the bus. These timeout values must be programmed into the TWIHS_SMBTR.
- Transmissions can optionally include a CRC byte, called Packet Error Check (PEC).
- A set of addresses have been reserved for protocol handling, such as alert response address (ARA) and host header (HH) address. Address matching on these addresses can be enabled by configuring the TWIHS_CR.

#### 45.6.5.6.1 Packet Error Checking

Each SMBus transfer can optionally end with a CRC byte, called the PEC byte. Writing a one to TWIHS_CR.PECEN will send/check the PEC field in the current transfer. The PEC generator is always updated on every bit transmitted or received, so that PEC handling on the following linked transfers is correct.

In Slave Receiver mode, the master calculates a PEC value and transmits it to the slave after all data bytes have been transmitted. Upon reception of this PEC byte, the slave compares it to the PEC value it has computed itself. If the values match, the data was received correctly, and the slave returns an ACK to the master. If the PEC values differ, data was corrupted, and the slave returns a NACK value. TWIHS_SR.PECERR is set automatically if a PEC error occurred.

In Slave Transmitter mode, the slave calculates a PEC value and transmits it to the master after all data bytes have been transmitted. Upon reception of this PEC byte, the master compares it to the PEC value it has computed itself. If the values match, the data was received correctly. If the PEC values differ, data was corrupted, and the master must take appropriate action.

See Slave Read Write Flowcharts for detailed flowcharts.

#### 45.6.5.6.2 Timeouts

The TWIHS SMBus Timing Register (TWIHS_SMBTR) configures the SMBus timeout values. If a timeout occurs, the slave leaves the bus. The TOUT bit is also set in TWIHS_SR.

#### 45.6.5.7 High-Speed Slave Mode

High-speed mode is enabled when a one is written to TWIHS_CR.HSEN. Furthermore, the analog pad filter must be enabled, a one must be written to TWIHS_FILTR.PADFEN and the FILT bit must be cleared. TWIHS High-speed mode operation is similar to TWIHS operation with the following exceptions:

1. A master code is received first at normal speed before entering High-speed mode period.
2. When TWIHS High-speed mode is active, clock stretching is only allowed after acknowledge (ACK), not-acknowledge (NACK), START (S) or REPEATED START (Sr) (as consequence OVF may happen).

TWIHS High-speed mode allows transfers of up to 3.4 Mbit/s.

The TWIHS slave in High-speed mode requires that the peripheral clock runs at a minimum of 14 MHz if slave clock stretching is enabled (SCLWSDIS bit at ‘0’). If slave clock stretching is disabled (SCLWSDIS bit at ‘1’), the peripheral clock must run at a minimum of 11 MHz (assuming the system has no latency).

**Note:** When slave clock stretching is disabled, the TWIHS_RHR must always be read before receiving the next data (MASTER write frame). It is strongly recommended to use either the polling method on the RXRDY flag in TWIHS_SR, or the DMA. If the receive is managed by an interrupt, the TWIHS interrupt priority must be set to the right level and its latency minimized to avoid receive overrun.

**Note:** When slave clock stretching is disabled, the TWIHS_THR must be filled with the first data to send before the beginning of the frame (MASTER read frame). It is strongly recommended to use either the polling method on the TXRDY flag in TWIHS_SR, or the DMA. If the transmit is managed by an interrupt, the TWIHS interrupt priority must be set to the right level and its latency minimized to avoid transmit underrun.

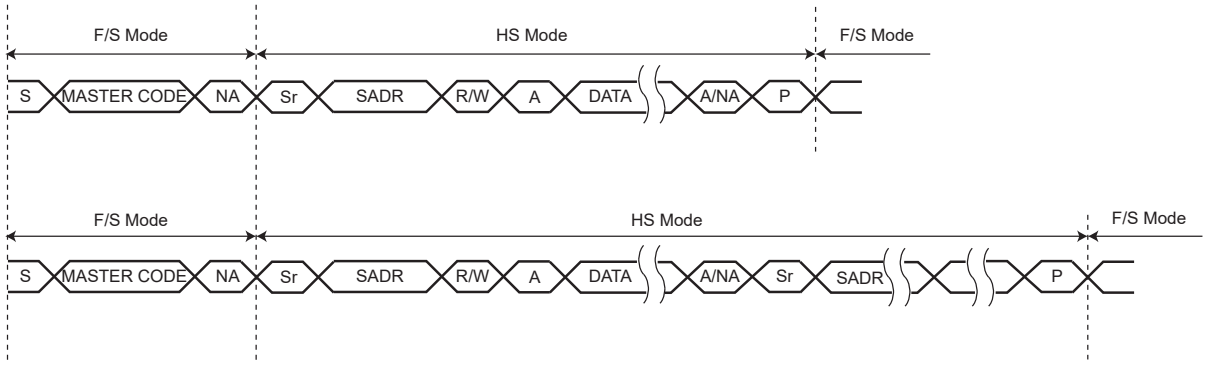
##### 45.6.5.7.1 Read/Write Operation

A TWIHS high-speed frame always begins with the following sequence:

1. START condition (S)
2. Master Code (0000 1XXX)
3. Not-acknowledge (NACK)

When the TWIHS is programmed in Slave mode and TWIHS High-speed mode is activated, master code matching is activated and internal timings are set to match the TWIHS High-speed mode requirements.

**Figure 45-38. High-Speed Mode Read/Write**




##### 45.6.5.7.2 Usage

TWIHS High-speed mode usage is the same as the standard TWIHS (See Read/Write Flowcharts).

#### 45.6.5.8 Alternative Command

In Slave mode, Alternative Command mode is useful when SMBus mode is enabled to send or check the PEC byte.

Alternative Command mode is enabled by setting the ACMEN bit of the TWIHS Control Register and the transfer is configured in TWIHS_ACR.

For a combined transfer with PEC, only TWIHS_ACR.NPEC must be set as the PEC byte is sent once at the end of the frame.

See Slave Read Write Flowcharts for detailed flowcharts.

#### 45.6.5.9 Asynchronous Partial Wakeup (SleepWalking)

The TWIHS includes an asynchronous start condition detector. It is capable of waking the device up from a Sleep mode upon an address match (and optionally an additional data match), including Sleep modes where the TWIHS peripheral clock is stopped.

After detecting the START condition on the bus, the TWIHS stretches TWCK until the TWIHS peripheral clock has started. The time required for starting the TWIHS depends on which Sleep mode the device is in. After the TWIHS peripheral clock has started, the TWIHS releases its TWCK stretching and receives one byte of data (slave address) on the bus. At this time, only a limited part of the device, including the TWIHS module, receives a clock, thus saving power. If the address phase causes a TWIHS address match (and, optionally, if the first data byte causes data match as well), the entire device is woken up and normal TWIHS address matching actions are performed. Normal TWIHS transfer then follows. If the TWIHS is not addressed (or if the optional data match fails), the TWIHS peripheral clock is automatically stopped and the device returns to its original Sleep mode.

The TWIHS has the capability to match on more than one address. The SADR1EN, SADR2EN and SADR3EN bits in TWIHS_SMR enable address matching on additional addresses which can be configured through SADR1, SADR2 and SADR3 fields in the TWIHS_SWMR. The SleepWalking matching process can be extended to the first received data byte if TWIHS_SMR.DATAMEN is set and, in this case, a complete matching includes address matching and first received data matching. TWIHS_SWMR.DATAM configures the data to match on the first received byte.

When the system is in Active mode and the TWIHS enters Asynchronous Partial Wakeup mode, the flag SVACC must be programmed as the unique source of the TWIHS interrupt and the data match comparison must be disabled.

When the system exits Wait mode as the result of a matching condition, the SVACC flag is used to determine if the TWIHS is the source of exit.

© 2018 Microchip Technology Inc.

**Figure 45-39. Address Match Only (Data Matching Disabled)**

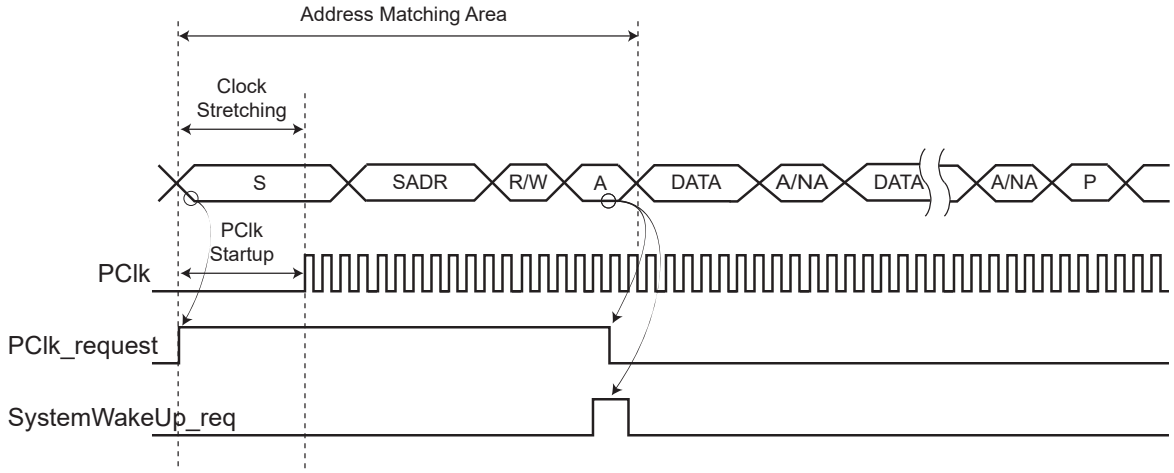



**Figure 45-40. No Address Match (Data Matching Disabled)**

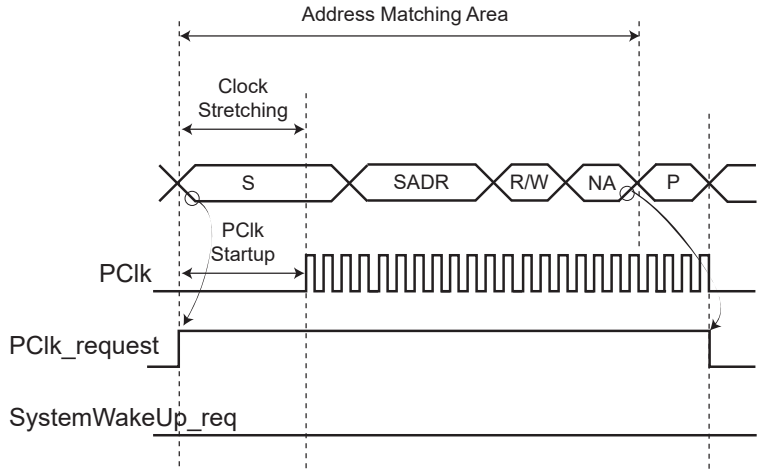



**Figure 45-41. Address Match and Data Match (Data Matching Enabled)**

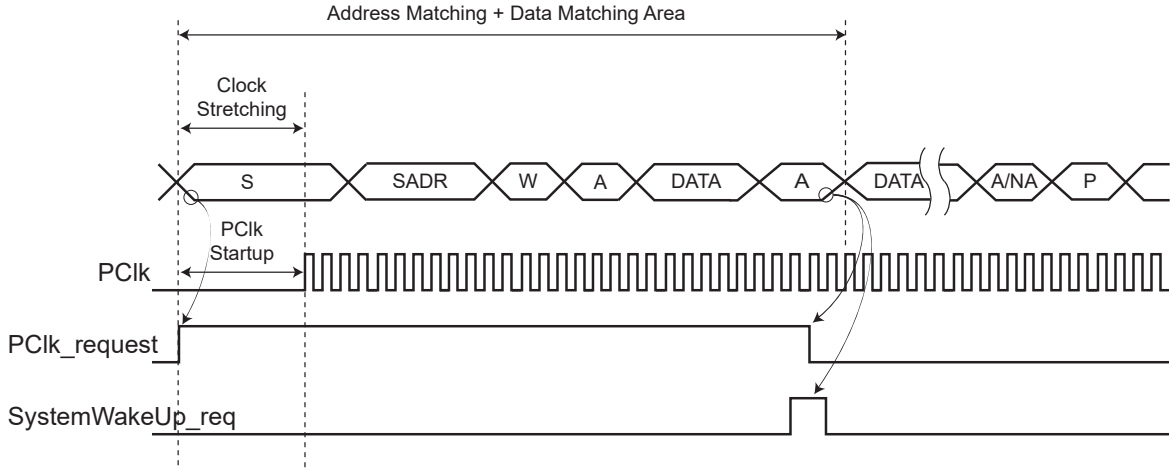
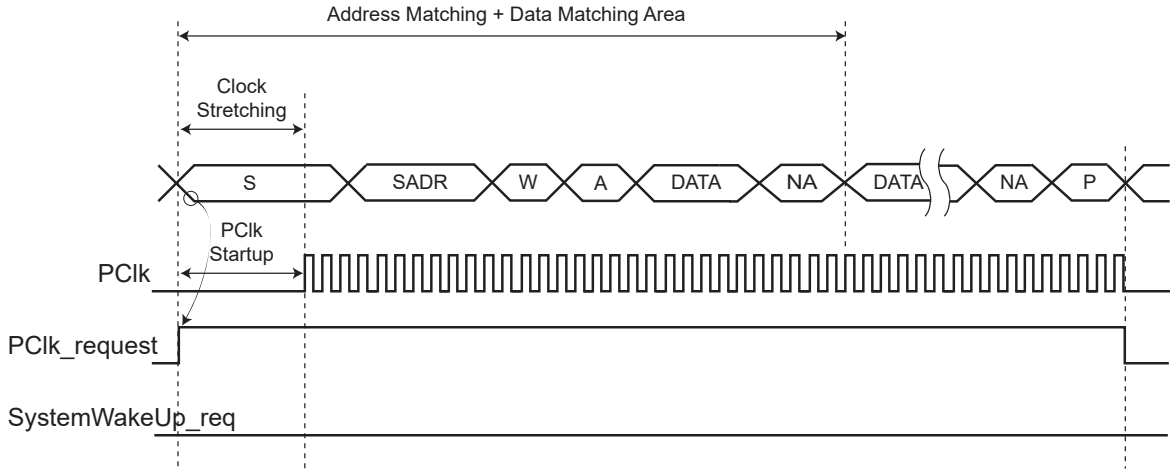


**Figure 45-42. Address Match and No Data Match (Data Matching Enabled)**




#### 45.6.5.10 Slave Read Write Flowcharts

The flowchart below illustrates an example of read and write operations in Slave mode. A polling or interrupt method can be used to check the status bits. The interrupt method requires that TWIHS_IER be configured first.

**Figure 45-43. Read Write Flowchart in Slave Mode**

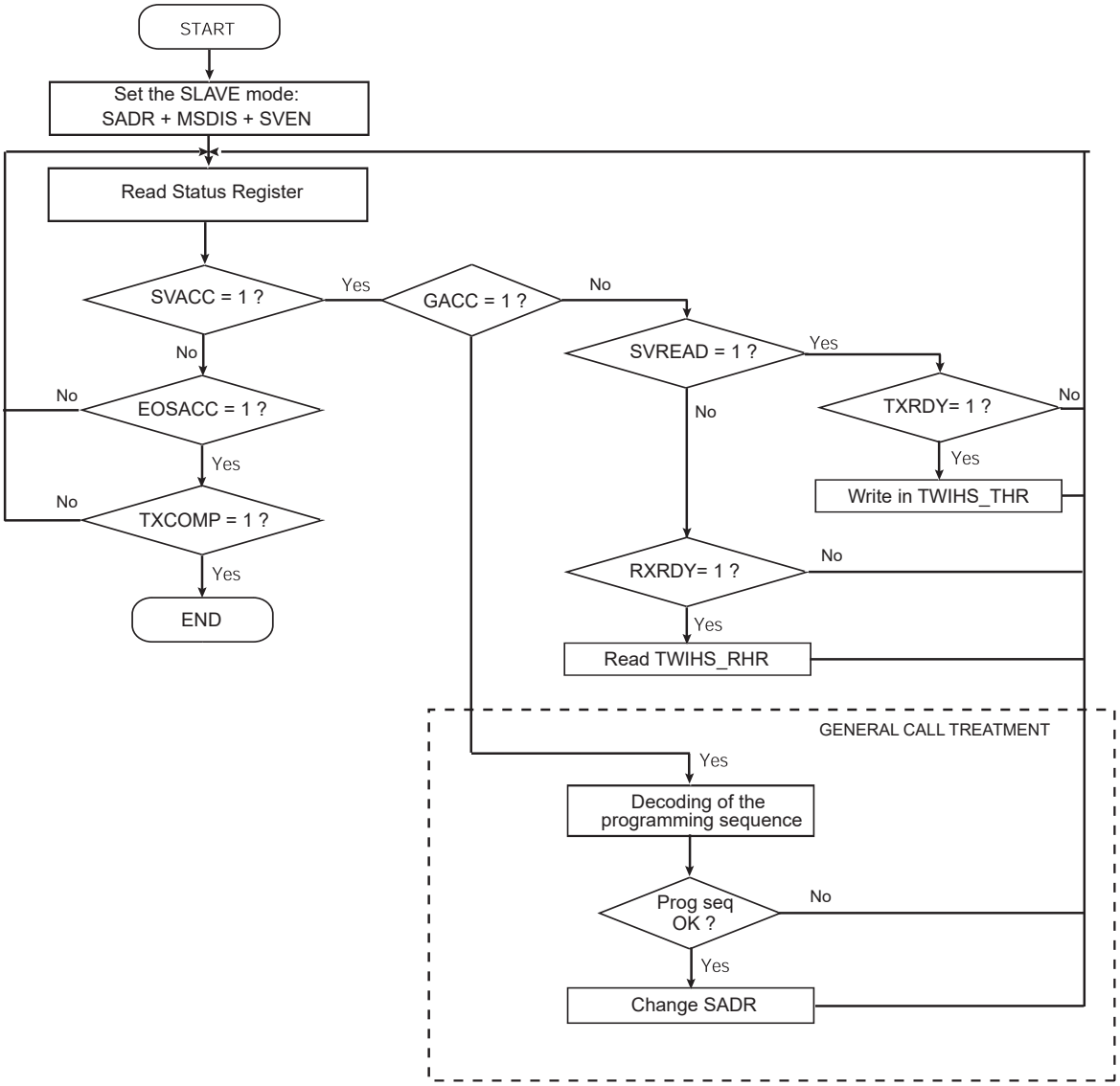


**Figure 45-44. Read Write Flowchart in Slave Mode with SMBus PEC**

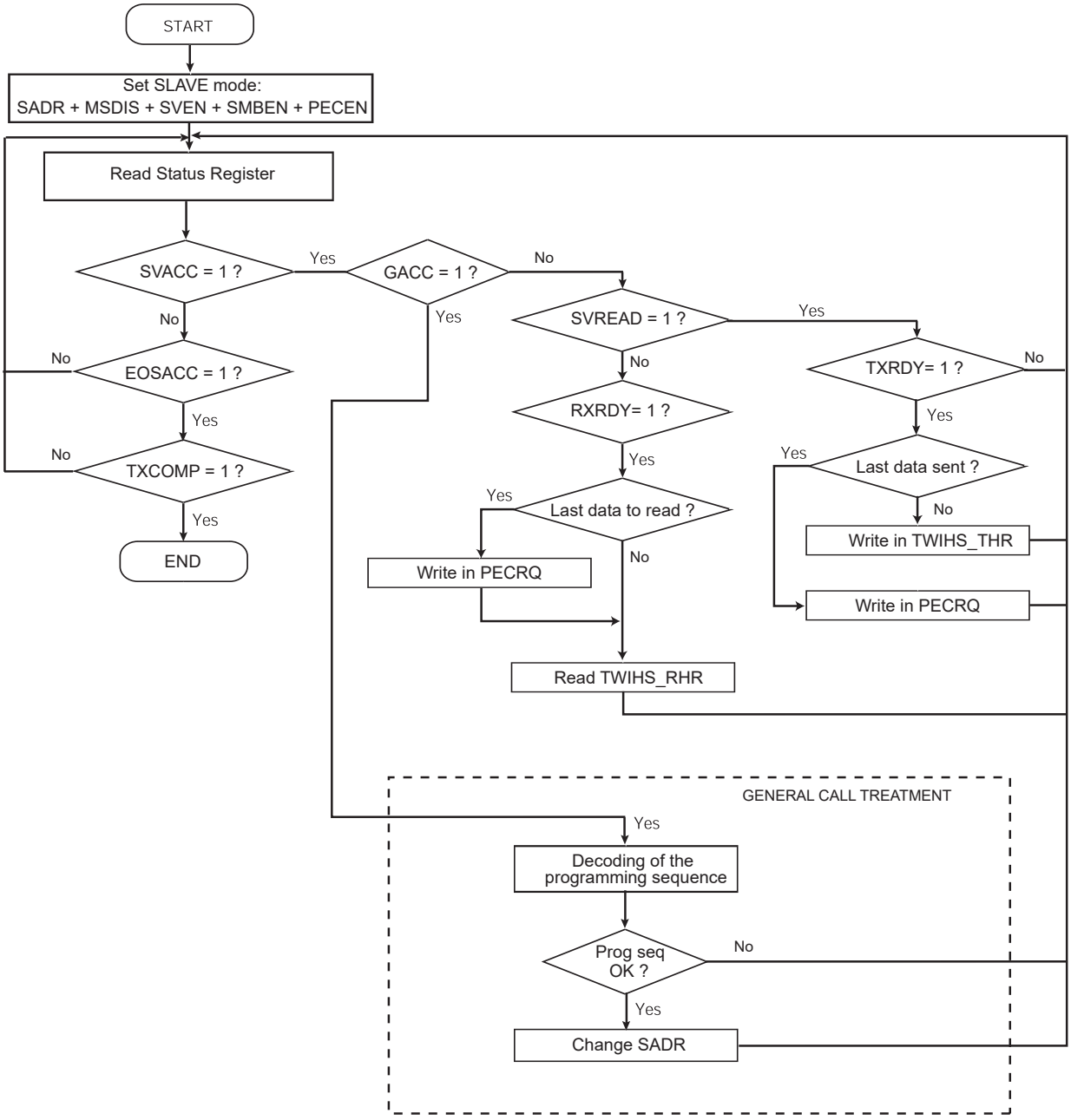
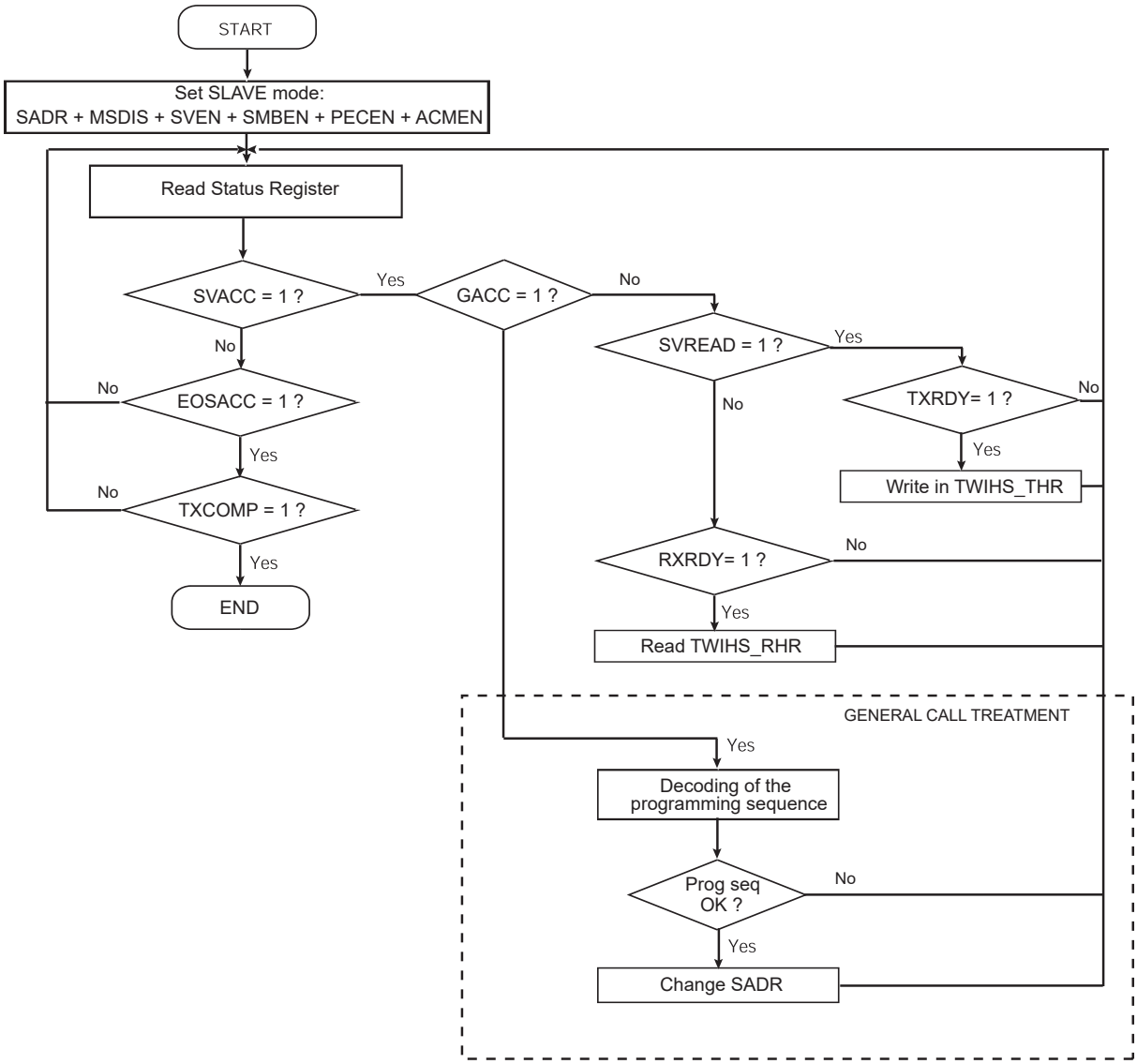


**Figure 45-45. Read Write Flowchart in Slave Mode with SMBus PEC and Alternative Command Mode**




### 45.6.6 FIFOs

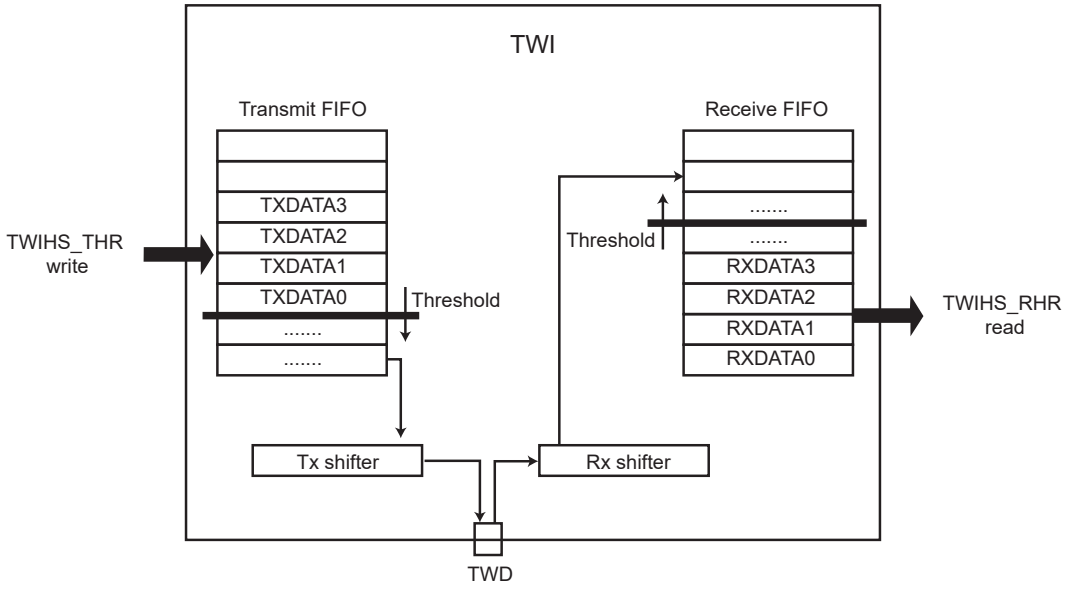
#### 45.6.6.1 Overview

The TWI includes two FIFOs which can be enabled/disabled using TWIHS_CR.FIFOEN/FIFODIS. Both Master and Slave modes must be disabled before enabling or disabling the FIFOs (TWIHS_CR.MSDIS/SVDIS).

Writing TWIHS_CR.FIFOEN to ‘1’ enables a 16-data Transmit FIFO and a 16-data Receive FIFO.

It is possible to write or to read single or multiple data in the same access to TWIHS_THR/RHR, depending onTWIHS_FMR.TXRDYM/RXRDYM settings.

**Figure 45-46. FIFOs Block Diagram**




### 45.6.6.2 Sending Data with FIFO Enabled

When the Transmit FIFO is enabled, write access to TWIHS_THR loads the Transmit FIFO.

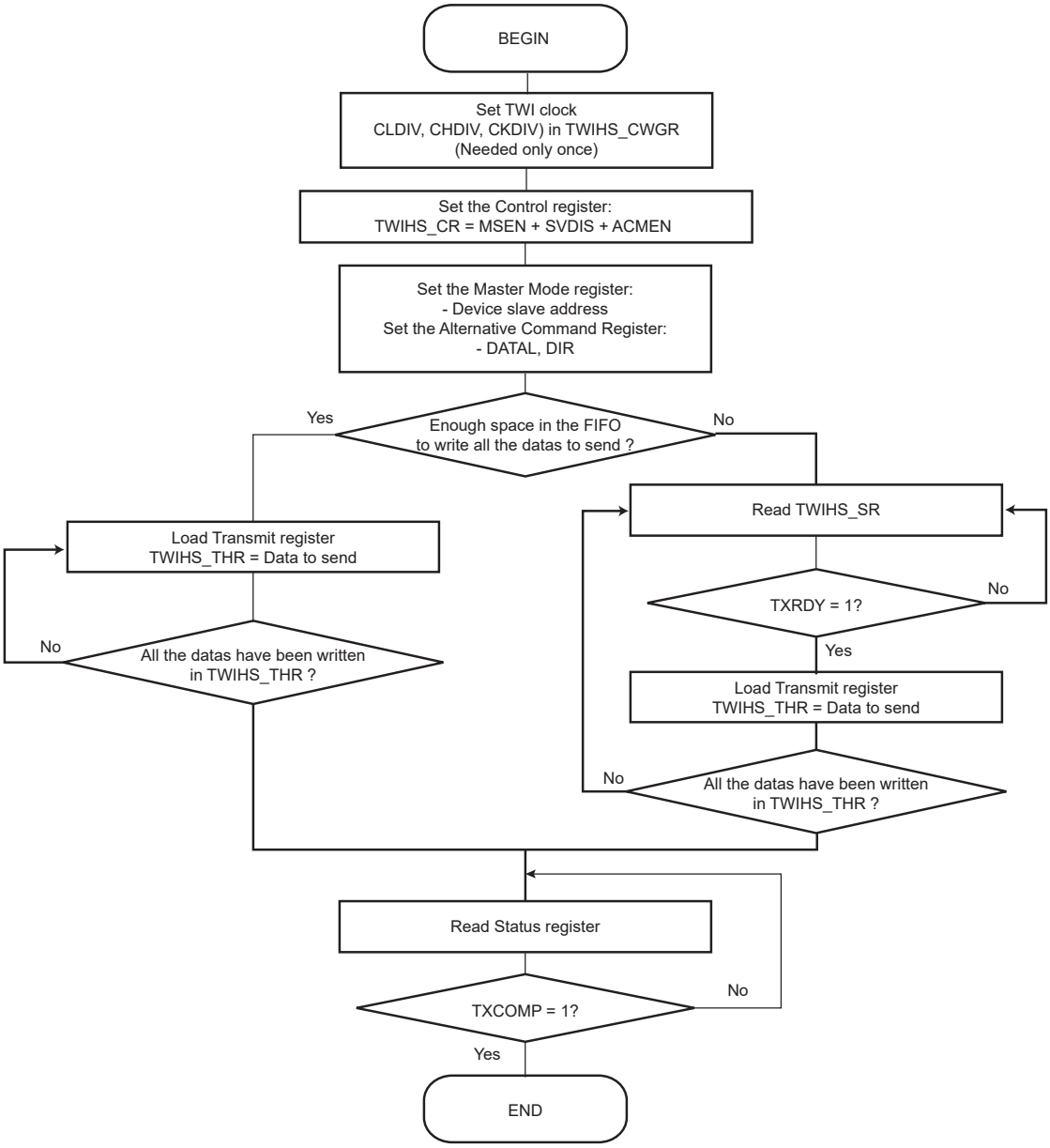
The Transmit FIFO level is provided in TWIHS_FLR.TXFL. If the FIFO can accept the number of data to be transmitted, there is no need to monitor TWIHS_SR.TXRDY and the data can be successively written in TWIHS_THR.

If the FIFO cannot accept the data due to insufficient space, wait for the TXRDY flag to be set before writing the data in TWIHS_THR.

When the space in the FIFO allows only a portion of the data to be written, the TXRDY flag must be monitored before writing the remaining data.

See the figure below and the figure Sending/Receiving Data with FIFO Enabled in Slave Mode.

**Figure 45-47. Sending Data with FIFO Enabled in Master Mode**



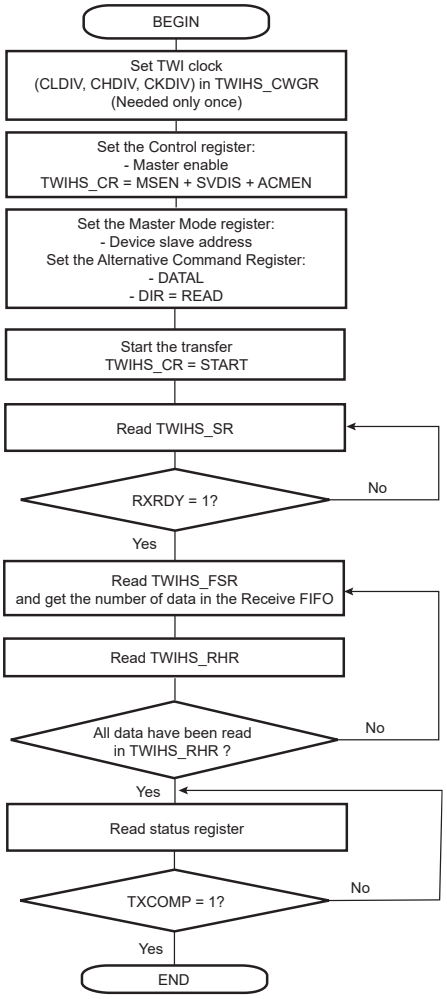

### 45.6.6.3 Receiving Data with FIFO Enabled

When the Receive FIFO is enabled, TWIHS_RHR access reads the FIFO.

When data are present in the Receive FIFO (RXRDY flag set to ‘1’), the exact number of data can be checked with TWIHS_FLR.RXFL. All the data can be read successively in TWIHS_RHR without checking the TWIHS_SR.RXRDY flag between each access.

See the figure below and the figure Sending/Receiving Data with FIFO Enabled in Slave Mode.

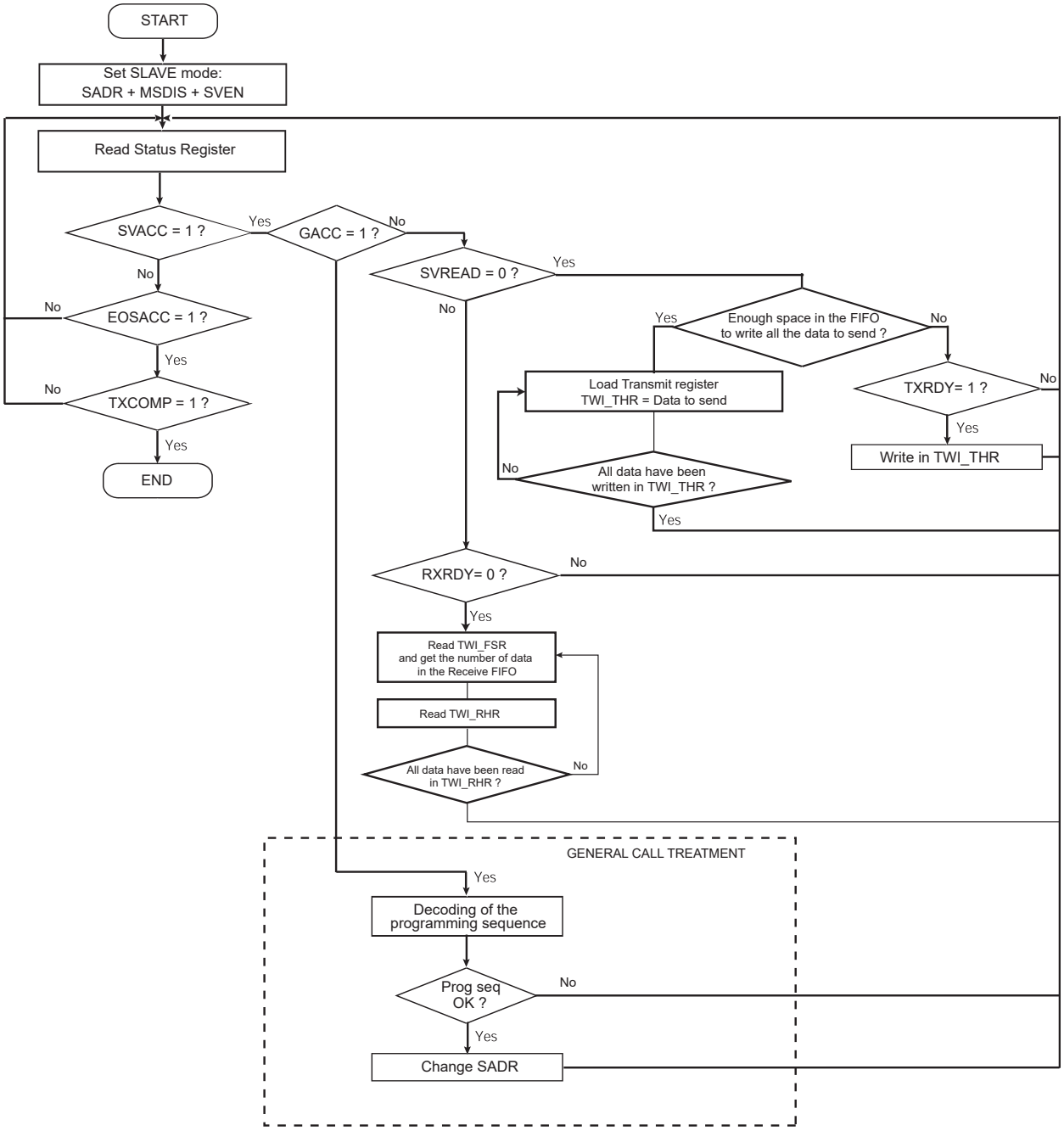
**Figure 45-48. Receiving Data with FIFO Enabled in Master Mode**




### 45.6.6.4 Sending/Receiving with FIFO Enabled in Slave Mode

See sections 45.6.6.2 Sending Data with FIFO Enabled and 45.6.6.3 Receiving Data with FIFO Enabled for details.

**Figure 45-49. Sending/Receiving Data with FIFO Enabled in Slave Mode**




### 45.6.6.5 Clearing/Flushing FIFOs

Each FIFO can be cleared/flushed using TWIHS_CR.TXFCLR/RXFCLR.

### 45.6.6.6 TXRDY and RXRDY Behavior

TWIHS_SR.TXRDY/RXRDY flags display a specific behavior when FIFOs are enabled.

TXRDY indicates if a data can be written in the Transmit FIFO. Thus the TXRDY flag is set as long as the Transmit FIFO can accept new data. See figure Figure 45-50.

RXRDY indicates if an unread data is present in the Receive FIFO. Thus the RXRDY flag is set as soon as one unread data is in the Receive FIFO. Refer to figure Figure 45-51.

TXRDY and RXRDY behavior can be modified using the TXRDYM and RXRDYM fields in the TWI FIFO Mode Register (TWIHS_FMR) to reduce the number of accesses to TWIHS_THR/RHR.

As an example, in Master mode, the Transmit FIFO can be loaded with multiple data in the same access by configuring TXRDYM>0.

See TWI FIFO Mode Register for the FIFO configuration.

**Figure 45-50. TXRDY Behavior when TXRDYM = 0 in Master Mode**

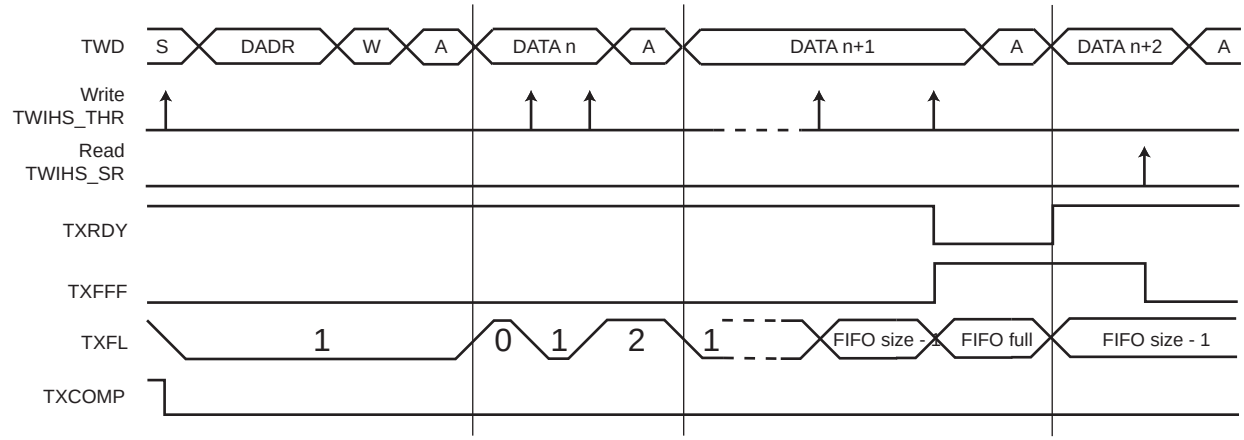



**Figure 45-51. RXRDY Behavior when RXRDYM = 0 in Master and Slave Modes**

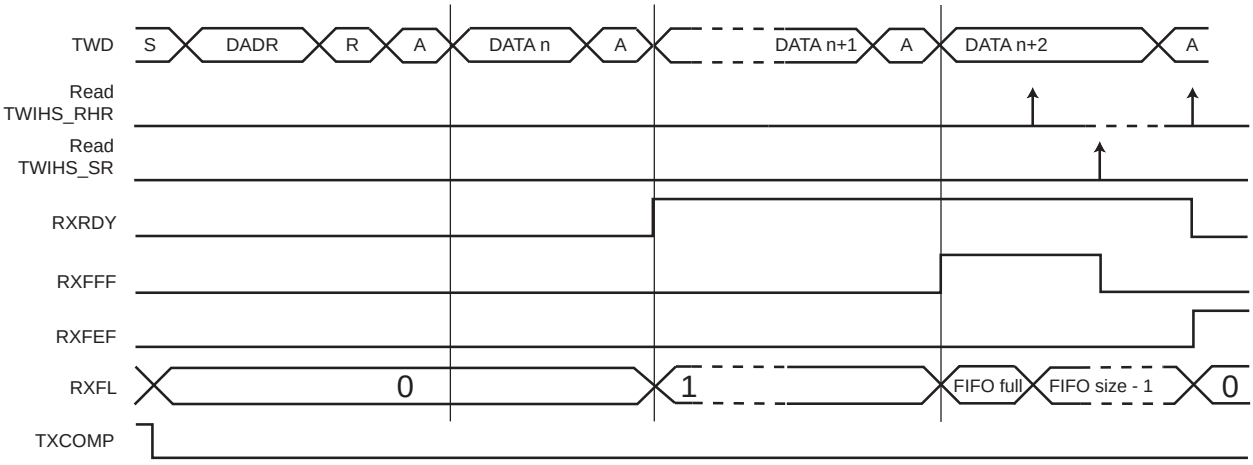
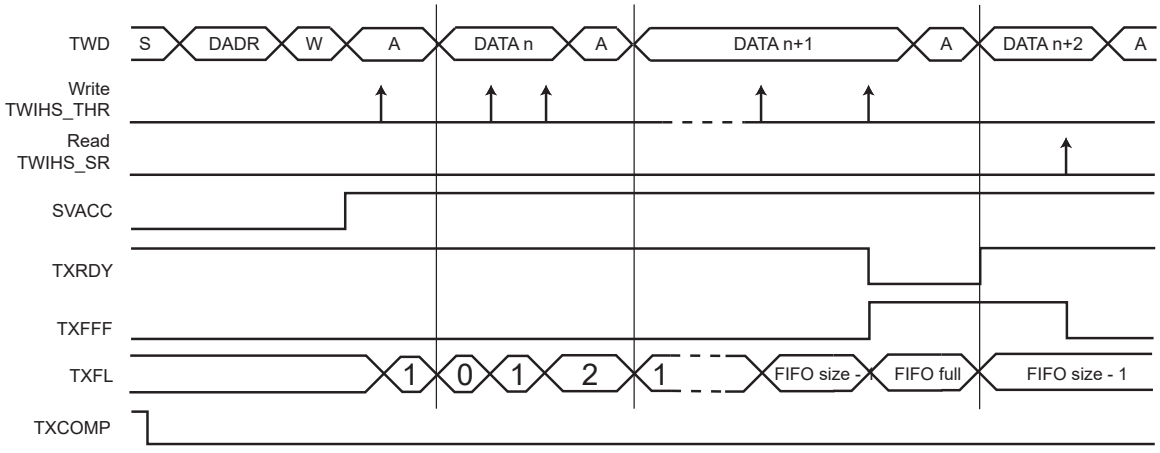



**Figure 45-52. TXRDY Behavior when TXRDYM = 0 in Slave Mode**



#### 45.6.6.7 Single Data Mode

In Single Data mode, only one data is written every time TWIHS_THR is accessed, and only one data is read every time TWIHS_RHR is accessed.

When TWIHS_FMR.TXRDYM = 0, the Transmit FIFO operates in Single Data mode.

When TWIHS_FMR.RXRDYM = 0, the Receive FIFO operates in Single Data mode.

See sections 45.7.18 TWIHS_THR and 45.7.15 TWIHS_RHR.

#### 45.6.6.8 Multiple Data Mode

Multiple Data mode minimizes the number of accesses by concatenating the data to send/read in one access.

When TWIHS_FMR.TXRDYM > 0, the Transmit FIFO operates in Multiple Data mode.

When TWIHS_FMR.RXRDYM > 0, the Receive FIFO operates in Multiple Data mode.

In Multiple Data mode, it is possible to write/read up to four data in one TWIHS_THR/TWIHS_RHR register access.

The number of data to write/read is defined by the size of the register access. If the access is a byte-size register access, only one data is written/read. If the access is a halfword size register access, then up to two data are read and only one data is written. Lastly, if the access is a wordsize register access, then up to four data are read and up to two data are written.

Written/Read data are always right-aligned, as described in sections 45.7.16 TWIHS_RHR (FIFO_ENABLED) and 45.7.19 TWIHS_THR (FIFO_ENABLED).

As an example, if the Transmit FIFO is empty and there are six data to send, either of the following write accesses may be performed:

- six TWIHS_THR-byte write accesses
- three TWIHS_THR-halfword write accesses
- one TWIHS_THR-word write access and one TWIHS_THR halfword write access

With a Receive FIFO containing six data, any of the following read accesses may be performed:

- six TWIHS_RHR-byte read accesses
- three TWIHS_RHR-halfword read accesses
- one TWIHS_RHR-word read access and one TWIHS_RHR-halfword read access

##### 45.6.6.8.1 TXRDY and RXRDY Configuration

In Multiple Data mode, it is possible to write one or more data in the same TWIHS_THR/TWIHS_RHR access. The TXRDY flag indicates if one or more data can be written in the FIFO depending on the configuration of TWIHS_FMR.TXRDYM/RXRDYM.

As an example, if two data are written each time in TWIHS_THR, it is useful to configure the TXRDYM field to the value ‘1’ so that the TXRDY flag is at ‘1’ only when at least two data can be written in the Transmit FIFO.

In the same way, if four data are read each time in TWIHS_RHR, it is useful to configure the RXRDYM field to the value ‘2’ so that the RXRDY flag is at ‘1’ only when at least four unread data are in the Receive FIFO.

##### 45.6.6.8.2 DMAC

When FIFOs operate in Multiple Data mode, the DMAC transfer type must be configured in byte, halfword or word depending on the TWIHS_FMR.TXRDYM/RXRDYM settings.

#### 45.6.6.9 Transmit FIFO Lock

If a frame is terminated early due to a not-acknowledge error (NACK flag), SMBus timeout error (TOUT flag) or master code acknowledge error (MACK flag), a lock is set on the Transmit FIFO preventing any new frame from being sent until it is cleared. This allows clearing the FIFO if needed, resetting DMAC channels, etc., without any risk.

TWIHS_SR.LOCK is used to check the state of the Transmit FIFO lock.

The Transmit FIFO lock can be cleared by setting TWIHS_CR.TXFLCLR to ‘1’.

#### 45.6.6.10 FIFO Pointer Error

A FIFO overflow is reported in TWIHS_FSR.

If the Transmit FIFO is full and a write access is performed on TWIHS_THR, it generates a Transmit FIFO pointer error and sets TWIHS_FSR.TXFPTEF.

In Multiple Data mode, if the number of data written in TWIHS_THR (according to the register access size) is greater than the free space in the Transmit FIFO, a Transmit FIFO pointer error is generated and TWIHS_FSR.TXFPTEF is set.

A FIFO underflow is reported in TWIHS_FSR.

In Multiple Data mode, if the number of data read in TWIHS_RHR (according to the register access size) is greater than the number of unread data in the Receive FIFO, a Receive FIFO pointer error is generated and TWIHS_FSR.RXFPTEF is set.

No pointer error occurs if the FIFO state/level is checked before writing/reading in TWIHS_THR/TWIHS_RHR. The FIFO state/level can be checked either with TXRDY, RXRDY, TXFL or RXFL. When a pointer error occurs, other FIFO flags may not behave as expected; their states should be ignored.

If a Transmit or Receive pointer error occurs, a software reset must be performed using TWIHS_CR.SWRST. Note that issuing a software while transmitting might leave a slave in an unknown state holding the TWD line. In such case, a Bus Clear Command will allow to make the slave release the TWD line (the first frame sent after might not be received properly by the slave).

#### 45.6.6.11 FIFO Thresholds

Each Transmit and Receive FIFO includes a threshold feature used to set a flag and an interrupt when a FIFO threshold is crossed. Thresholds are defined as a number of data in the FIFO, and the FIFO state (TXFL or RXFL) represents the number of data currently in the FIFO.

The Transmit FIFO threshold can be set using the field TWIHS_FMR.TXFTHRES. Each time the Transmit FIFO level goes from ‘above threshold’ to ‘equal to or below threshold’, the flag TWIHS_FESR.TXFTHF is set. The application is warned that the Transmit FIFO has reached the defined threshold and that it can be reloaded.

The Receive FIFO threshold can be set using the field TWIHS_FMR.RXFTHRES. Each time the Receive FIFO level goes from ‘below threshold’ to ‘equal to or above threshold’, the flag TWIHS_FESR.RXFTHF is set. The application is warned that the Receive FIFO has reached the defined threshold and that it can be read to prevent an underflow.

The TXFTHF and RXFTHF flags can be configured to generate an interrupt using TWIHS_FIER and TWIHS_FIDR.

#### 45.6.6.12 FIFO Flags

FIFOs come with a set of flags which can be configured to generate interrupts through TWIHS_FIER and TWIHS_FIDR.

FIFO flags state can be read in TWIHS_FSR. They are cleared when TWIHS_FSR is read.

### 45.6.7 TWIHS Comparison Function on Received Character

The comparison function differs if asynchronous partial wakeup (SleepWalking) is enabled or not.

If asynchronous partial wakeup is disabled (see the section “Power Management Controller (PMC)”), the TWIHS can extend the address matching on up to three slave addresses. The SADR1EN, SADR2EN and SADR3EN bits in TWIHS_SMR enable address matching on additional addresses which can be configured through SADR1, SADR2 and SADR3 fields in the TWIHS_SWMR. The DATAMEN bit in the TWIHS_SMR has no effect.

The SVACC bit is set when there is a comparison match with the received slave address.

### 45.6.8 Register Write Protection

To prevent any single software error from corrupting TWIHS behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the TWIHS Write Protection Mode Register (TWIHS_WPMR).

If a write access to a write-protected register is detected, the WPVS bit in the TWIHS Write Protection Status Register (TWIHS_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading TWIHS_WPSR.

The following registers can be write-protected:

- TWIHS Slave Mode Register
- TWIHS Clock Waveform Generator Register
- TWIHS SMBus Timing Register
- TWIHS SleepWalking Matching Register

## 45.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | TWIHS_CR | 7:0 | SWRST | QUICK | SVDIS | SVEN | MSDIS | MSEN | STOP | START |
| | | 15:8 | CLEAR | PECRQ | PECDIS | PECEN | SMBDIS | SMBEN | HSDIS | HSEN |
| | | 23:16 | | | | | | | ACMDIS | ACMEN |
| | | 31:24 | | | FIFODIS | FIFOEN | | LOCKCLR | | THRCLR |
| 0x00 | TWIHS_CR (FIFO_ENABLED) | 7:0 | SWRST | QUICK | SVDIS | SVEN | MSDIS | MSEN | STOP | START |
| | | 15:8 | CLEAR | PECRQ | PECDIS | PECEN | SMBDIS | SMBEN | HSDIS | HSEN |
| | | 23:16 | | | | | | | ACMDIS | ACMEN |
| | | 31:24 | | | FIFODIS | FIFOEN | | TXFLCLR | RXFCLR | TXFCLR |
| 0x04 | TWIHS_MMR | 7:0 | | | | | | | | |
| | | 15:8 | | | | MREAD | | | IADRSZ[1:0] | |
| | | 23:16 | | DADR[6:0] | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | TWIHS_SMR | 7:0 | | SCLWSDIS | | | SMHH | SMDA | | NACKEN |
| | | 15:8 | | MASK[6:0] | | | | | | |
| | | 23:16 | | SADR[6:0] | | | | | | |
| | | 31:24 | DATAMEN | SADR3EN | SADR2EN | SADR1EN | | | | |
| 0x0C | TWIHS_IADR | 7:0 | IADR[7:0] | | | | | | | |
| | | 15:8 | IADR[15:8] | | | | | | | |
| | | 23:16 | IADR[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | TWIHS_CWGR | 7:0 | CLDIV[7:0] | | | | | | | |
| | | 15:8 | CHDIV[7:0] | | | | | | | |
| | | 23:16 | | | | CKSRC | | CKDIV[2:0] | | |
| | | 31:24 | | | | HOLD[4:0] | | | | |
| 0x14 ... 0x1F | Reserved | | | | | | | | | |
| 0x20 | TWIHS_SR | 7:0 | UNRE | OVRE | GACC | SVACC | SVREAD | TXRDY | RXRDY | TXCOMP |
| | | 15:8 | | | | | EOSACC | SCLWS | ARBLST | NACK |
| | | 23:16 | LOCK | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| | | 31:24 | | | | | | | SDA | SCL |
| 0x20 | TWIHS_SR (FIFO_ENABLED) | 7:0 | UNRE | OVRE | GACC | SVACC | SVREAD | TXRDY | RXRDY | TXCOMP |
| | | 15:8 | | | | | EOSACC | SCLWS | ARBLST | NACK |
| | | 23:16 | TXFLOCK | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| | | 31:24 | | | | | | | SDA | SCL |
| 0x24 | TWIHS_IER | 7:0 | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| | | 15:8 | | | | | EOSACC | SCL_WS | ARBLST | NACK |
| | | 23:16 | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| | | 31:24 | | | | | | | | |
| 0x28 | TWIHS_IDR | 7:0 | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| | | 15:8 | | | | | EOSACC | SCL_WS | ARBLST | NACK |
| | | 23:16 | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x2C | TWIHS_IMR | 7:0 | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| | | 15:8 | | | | | EOSACC | SCL_WS | ARBLST | NACK |
| | | 23:16 | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| | | 31:24 | | | | | | | | |
| 0x30 | TWIHS_RHR | 7:0 | RXDATA[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x30 | TWIHS_RHR (FIFO_ENABLED) | 7:0 | RXDATA0[7:0] | | | | | | | |
| | | 15:8 | RXDATA1[7:0] | | | | | | | |
| | | 23:16 | RXDATA2[7:0] | | | | | | | |
| | | 31:24 | RXDATA3[7:0] | | | | | | | |
| 0x34 | TWIHS_THR | 7:0 | TXDATA[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x34 | TWIHS_THR (FIFO_ENABLED) | 7:0 | TXDATA0[7:0] | | | | | | | |
| | | 15:8 | TXDATA1[7:0] | | | | | | | |
| | | 23:16 | TXDATA2[7:0] | | | | | | | |
| | | 31:24 | TXDATA3[7:0] | | | | | | | |
| 0x38 | TWIHS_SMBTR | 7:0 | | | | | PRESC[3:0] | | | |
| | | 15:8 | TLOWS[7:0] | | | | | | | |
| | | 23:16 | TLOWM[7:0] | | | | | | | |
| | | 31:24 | THMAX[7:0] | | | | | | | |
| 0x3C ... 0x3F | Reserved | | | | | | | | | |
| 0x40 | TWIHS_ACR | 7:0 | DATAL[7:0] | | | | | | | |
| | | 15:8 | | | | | | | PEC | DIR |
| | | 23:16 | NDATAL[7:0] | | | | | | | |
| | | 31:24 | | | | | | | NPEC | NDIR |
| 0x44 | TWIHS_FILTR | 7:0 | | | | | | PADFCFG | PADFEN | FILT |
| | | 15:8 | | | | | | THRES[2:0] | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x48 ... 0x4B | Reserved | | | | | | | | | |
| 0x4C | TWIHS_SWMR | 7:0 | | SADR1[6:0] | | | | | | |
| | | 15:8 | | SADR2[6:0] | | | | | | |
| | | 23:16 | | SADR3[6:0] | | | | | | |
| | | 31:24 | DATAM[7:0] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x50 | TWIHS_FMR | 7:0 | | | RXRDYM[1:0] | | | | TXRDYM[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | TXFTHRES[5:0] | | | | | |
| | | 31:24 | | | RXFTHRES[5:0] | | | | | |
| 0x54 | TWIHS_FLR | 7:0 | | | TXFL[5:0] | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | RXFL[5:0] | | | | | |
| | | 31:24 | | | | | | | | |
| 0x58 ... 0x5F | Reserved | | | | | | | | | |
| 0x60 | TWIHS_FSR | 7:0 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x64 | TWIHS_FIER | 7:0 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x68 | TWIHS_FIDR | 7:0 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x6C | TWIHS_FIMR | 7:0 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x70 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | TWIHS_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | TWIHS_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | WPVSRC[23:16] | | | | | | | |

### 45.7.1 TWIHS Control Register

**Name:** TWIHS_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | FIFODIS | FIFOEN | | LOCKCLR | | THRCLR |
| Access | | | W | W | | W | | W |
| Reset | | | – | – | | – | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ACMDIS | ACMEN |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CLEAR | PECRQ | PECDIS | PECEN | SMBDIS | SMBEN | HSDIS | HSEN |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWRST | QUICK | SVDIS | SVEN | MSDIS | MSEN | STOP | START |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 29 – FIFODIS** FIFO Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Transmit and Receive FIFOs. |

**Bit 28 – FIFOEN** FIFO Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Transmit and Receive FIFOs. |

**Bit 26 – LOCKCLR** Lock Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears the TWIHS FSM lock. |

**Bit 24 – THRCLR** Transmit Holding Register Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears the Transmit Holding Register and sets TXRDY, TXCOMP flags. |

**Bit 17 – ACMDIS** Alternative Command Mode Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Alternative Command mode disabled. |

**Bit 16 – ACMEN** Alternative Command Mode Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Alternative Command mode enabled. |

**Bit 15 – CLEAR** Bus CLEAR Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If Master mode is enabled, sends a bus clear command. |

**Bit 14 – PECRQ** PEC Request

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | A PEC check or transmission is requested. |

**Bit 13 – PECDIS** Packet Error Checking Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | SMBus PEC (CRC) generation and check disabled. |

**Bit 12 – PECEN** Packet Error Checking Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | SMBus PEC (CRC) generation and check enabled. |

**Bit 11 – SMBDIS** SMBus Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | SMBus mode disabled. |

**Bit 10 – SMBEN** SMBus Mode Enabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If SMBDIS = 0, SMBus mode enabled. |

**Bit 9 – HSDIS** TWIHS High-Speed Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | High-speed mode disabled. |

**Bit 8 – HSEN** TWIHS High-Speed Mode Enabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | High-speed mode enabled. |

**Bit 7 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Equivalent to a system reset. |

**Bit 6 – QUICK** SMBus Quick Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If Master mode is enabled, a SMBus Quick Command is sent. |

**Bit 5 – SVDIS** TWIHS Slave Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The Slave mode is disabled. The shifter and holding characters (if it contains data) are transmitted in case of read operation. In write operation, the character being transferred must be completely received before disabling. |

**Bit 4 – SVEN** TWIHS Slave Mode Enabled
Switching from Master to Slave mode is only permitted when TXCOMP = 1.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Slave mode (SVDIS must be written to 0). |

**Bit 3 – MSDIS** TWIHS Master Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The Master mode is disabled, all pending data is transmitted. The shifter and holding characters (if it contains data) are transmitted in case of write operation. In read operation, the character being transferred must be completely received before disabling. |

**Bit 2 – MSEN** TWIHS Master Mode Enabled
Switching from Slave to Master mode is only permitted when TXCOMP = 1.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Master mode (MSDIS must be written to 0). |

**Bit 1 – STOP** Send a STOP Condition

| Value | Description |
|---|---|
| 0 | No effect. |

| Value | Description |
|---|---|
| 1 | STOP condition is sent just after completing the current byte transmission in Master Read mode.<br>• In single data byte master read, both START and STOP must be set.<br>• In multiple data bytes master read, the STOP must be set after the last data received but one.<br>• In Master Read mode, if a NACK bit is received, the STOP is automatically performed.<br>• In master data write operation, a STOP condition will be sent after the transmission of the current data is finished. |

**Bit 0 – START** Send a START Condition
This action is necessary when the TWIHS peripheral needs to read data from a slave. When configured in Master mode with a write operation, a frame is sent as soon as the user writes a character in the Transmit Holding Register (TWIHS_THR).

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | A frame beginning with a START bit is transmitted according to the features defined in the TWIHS Master Mode Register (TWIHS_MMR). |

### 45.7.2 TWIHS Control Register (FIFO_ENABLED)

**Name:** TWIHS_CR (FIFO_ENABLED)
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

This configuration is relevant only if TWIHS.CR.FIFOEN = ‘1’.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | FIFODIS | FIFOEN | | TXFLCLR | RXFCLR | TXFCLR |
| Access | | | W | W | | W | W | W |
| Reset | | | – | – | | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ACMDIS | ACMEN |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CLEAR | PECRQ | PECDIS | PECEN | SMBDIS | SMBEN | HSDIS | HSEN |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWRST | QUICK | SVDIS | SVEN | MSDIS | MSEN | STOP | START |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 29 – FIFODIS** FIFO Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Transmit and Receive FIFOs. |

**Bit 28 – FIFOEN** FIFO Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Transmit and Receive FIFOs. |

**Bit 26 – TXFLCLR** Transmit FIFO Lock CLEAR

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears the Transmit FIFO Lock. |

**Bit 25 – RXFCLR** Receive FIFO Clear

| Value | Description |
|---|---|
| 0 | No effect. |

| Value | Description |
|---|---|
| 1 | Clears the Receive FIFO, Receive FIFO will become empty. |

**Bit 24 – TXFCLR** Transmit FIFO Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears the Transmit FIFO, Transmit FIFO will become empty. |

**Bit 17 – ACMDIS** Alternative Command Mode Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Alternative Command mode disabled. |

**Bit 16 – ACMEN** Alternative Command Mode Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Alternative Command mode enabled. |

**Bit 15 – CLEAR** Bus CLEAR Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If Master mode is enabled, sends a bus clear command. |

**Bit 14 – PECRQ** PEC Request

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | A PEC check or transmission is requested. |

**Bit 13 – PECDIS** Packet Error Checking Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | SMBus PEC (CRC) generation and check disabled. |

**Bit 12 – PECEN** Packet Error Checking Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | SMBus PEC (CRC) generation and check enabled. |

**Bit 11 – SMBDIS** SMBus Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | SMBus mode disabled. |

**Bit 10 – SMBEN** SMBus Mode Enabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If SMBDIS = 0, SMBus mode enabled. |

**Bit 9 – HSDIS** TWIHS High-Speed Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | High-speed mode disabled. |

**Bit 8 – HSEN** TWIHS High-Speed Mode Enabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | High-speed mode enabled. |

**Bit 7 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Equivalent to a system reset. |

**Bit 6 – QUICK** SMBus Quick Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If Master mode is enabled, a SMBus Quick Command is sent. |

**Bit 5 – SVDIS** TWIHS Slave Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The Slave mode is disabled. The shifter and holding characters (if it contains data) are transmitted in case of read operation. In write operation, the character being transferred must be completely received before disabling. |

**Bit 4 – SVEN** TWIHS Slave Mode Enabled
Switching from Master to Slave mode is only permitted when TXCOMP = 1.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Slave mode (SVDIS must be written to 0). |

**Bit 3 – MSDIS** TWIHS Master Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The Master mode is disabled, all pending data is transmitted. The shifter and holding characters (if it contains data) are transmitted in case of write operation. In read operation, the character being transferred must be completely received before disabling. |

**Bit 2 – MSEN** TWIHS Master Mode Enabled
Switching from Slave to Master mode is only permitted when TXCOMP = 1.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Master mode (MSDIS must be written to 0). |

**Bit 1 – STOP** Send a STOP Condition

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | STOP condition is sent just after completing the current byte transmission in Master Read mode.<br>• In single data byte master read, both START and STOP must be set.<br>• In multiple data bytes master read, the STOP must be set after the last data received but one.<br>• In Master Read mode, if a NACK bit is received, the STOP is automatically performed.<br>• In master data write operation, a STOP condition will be sent after the transmission of the current data is finished. |

**Bit 0 – START** Send a START Condition
This action is necessary when the TWIHS peripheral needs to read data from a slave. When configured in Master mode with a write operation, a frame is sent as soon as the user writes a character in the Transmit Holding Register (TWIHS_THR).

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | A frame beginning with a START bit is transmitted according to the features defined in the TWIHS Master Mode Register (TWIHS_MMR). |

### 45.7.3 TWIHS Master Mode Register

**Name:** TWIHS_MMR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | DADR[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | MREAD | | | IADRSZ[1:0] | |
| Access | | | | R/W | | | R/W | R/W |
| Reset | | | | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 22:16 – DADR[6:0]** Device Address
The device address is used to access slave devices in Read or Write mode. These bits are only used in Master mode.

**Bit 12 – MREAD** Master Read Direction

| Value | Description |
|---|---|
| 0 | Master write direction. |
| 1 | Master read direction. |

**Bits 9:8 – IADRSZ[1:0]** Internal Device Address Size

| Value | Name | Description |
|---|---|---|
| 0 | NONE | No internal device address |
| 1 | 1_BYTE | One-byte internal device address |
| 2 | 2_BYTE | Two-byte internal device address |
| 3 | 3_BYTE | Three-byte internal device address |

### 45.7.4 TWIHS Slave Mode Register

**Name:** TWIHS_SMR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TWIHS Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATAMEN | SADR3EN | SADR2EN | SADR1EN | | | | |
| Access | R/W | R/W | R/W | R/W | | | | |
| Reset | 0 | 0 | 0 | 0 | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | SADR[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | MASK[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | SCLWSDIS | | | SMHH | SMDA | | NACKEN |
| Access | | R/W | | | R/W | R/W | | R/W |
| Reset | | 0 | | | 0 | 0 | | 0 |

**Bit 31 – DATAMEN** Data Matching Enable

| Value | Description |
|---|---|
| 0 | Data matching on first received data is disabled. |
| 1 | Data matching on first received data is enabled. |

**Bit 30 – SADR3EN** Slave Address 3 Enable

| Value | Description |
|---|---|
| 0 | Slave address 3 matching is disabled. |
| 1 | Slave address 3 matching is enabled. |

**Bit 29 – SADR2EN** Slave Address 2 Enable

| Value | Description |
|---|---|
| 0 | Slave address 2 matching is disabled. |
| 1 | Slave address 2 matching is enabled. |

**Bit 28 – SADR1EN** Slave Address 1 Enable

| Value | Description |
|---|---|
| 0 | Slave address 1 matching is disabled. |

| Value | Description |
|---|---|
| 1 | Slave address 1 matching is enabled. |

**Bits 22:16 – SADR[6:0]** Slave Address
The slave device address is used in Slave mode in order to be accessed by master devices in Read or Write mode.

SADR must be programmed before enabling the Slave mode or after a general call. Writes at other times have no effect.

**Bits 14:8 – MASK[6:0]** Slave Address Mask
A mask can be applied on the slave device address in Slave mode in order to allow multiple address answer. For each bit of the MASK field set to 1, the corresponding SADR bit is masked.

If the MASK field value is 0, no mask is applied to the SADR field.

**Bit 6 – SCLWSDIS** Clock Wait State Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clock stretching disabled in Slave mode, OVRE and UNRE indicate an overrun/underrun. |

**Bit 3 – SMHH** SMBus Host Header

| Value | Description |
|---|---|
| 0 | Acknowledge of the SMBus host header disabled. |
| 1 | Acknowledge of the SMBus host header enabled. |

**Bit 2 – SMDA** SMBus Default Address

| Value | Description |
|---|---|
| 0 | Acknowledge of the SMBus default address disabled. |
| 1 | Acknowledge of the SMBus default address enabled. |

**Bit 0 – NACKEN** Slave Receiver Data Phase NACK enable

| Value | Description |
|---|---|
| 0 | Normal value to be returned in the ACK cycle of the data phase in Slave Receiver mode. |
| 1 | NACK value to be returned in the ACK cycle of the data phase in Slave Receiver mode. |

### 45.7.5 TWIHS Internal Address Register

**Name:** TWIHS_IADR
**Offset:** 0x0C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IADR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IADR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IADR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – IADR[23:0]** Internal Address
0, 1, 2 or 3 bytes depending on IADRSZ.

### 45.7.6 TWIHS Clock Waveform Generator Register

**Name:** TWIHS_CWGR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TWIHS Write Protection Mode Register.

TWIHS_CWGR is used in Master mode only.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | HOLD[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | | CKSRC | | CKDIV[2:0] | | |
| Access | | | | R/W | | R/W | R/W | R/W |
| Reset | | | | 0 | | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | CHDIV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CLDIV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 28:24 – HOLD[4:0]** TWD Hold Time Versus TWCK Falling
If High-speed mode is selected TWD is internally modified on the TWCK falling edge to meet the I2C specified maximum hold time, else if High-speed mode is not configured TWD is kept unchanged after TWCK falling edge for a period of (HOLD + 3) × $t_{peripheral\ clock}$.

**Bit 20 – CKSRC** Transfer Rate Clock Source

| Value | Name | Description |
|---|---|---|
| 0 | PERIPH_CK | Peripheral clock is used to generate the TWIHS baud rate. |
| 1 | GCLK | GCLK is used to generate the TWIHS baud rate. |

**Bits 18:16 – CKDIV[2:0]** Clock Divider
The CKDIV is used to increase both SCL high and low periods.

**Bits 15:8 – CHDIV[7:0]** Clock High Divider
The SCL high period is defined as follows:

If CKSRC = 0

$t_{high} = ((CHDIV \times 2^{CKDIV}) + 3) \times t_{peripheral\ clock}$

If CKSRC = 1

$t_{high} = (CHDIV \times 2^{CKDIV}) \times t_{external\ clock}$

**Bits 7:0 – CLDIV[7:0]** Clock Low Divider
The SCL low period is defined as follows:

If CKSRC = 0

$t_{low} = ((CLDIV \times 2^{CKDIV}) + 3) \times t_{peripheral\ clock}$

If CKSRC = 1

$t_{low} = (CLDIV \times 2^{CKDIV}) \times t_{external\ clock}$

### 45.7.7 TWIHS Status Register

**Name:** TWIHS_SR
**Offset:** 0x20
**Reset:** 0x03000009
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | SDA | SCL |
| Access | | | | | | | R | R |
| Reset | | | | | | | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | LOCK | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| Access | R | | R | R | R | R | | R |
| Reset | 0 | | 0 | 0 | 0 | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOSACC | SCLWS | ARBLST | NACK |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | GACC | SVACC | SVREAD | TXRDY | RXRDY | TXCOMP |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |

**Bit 25 – SDA** SDA Line Value

| Value | Description |
|---|---|
| 0 | SDA line sampled value is ‘0’. |
| 1 | SDA line sampled value is ‘1’. |

**Bit 24 – SCL** SCL Line Value

| Value | Description |
|---|---|
| 0 | SCL line sampled value is ‘0’. |
| 1 | SCL line sampled value is ‘1.’ |

**Bit 23 – LOCK** TWIHS Lock due to Frame Errors (cleared by writing a one to bit LOCKCLR in TWIHS_CR)

| Value | Description |
|---|---|
| 0 | The TWIHS is not locked or LOCKCLR command issued in TWIHS_CR. |
| 1 | The TWIHS is locked due to frame errors (see Handling Errors in Alternative Command). |

**Bit 21 – SMBHHM** SMBus Host Header Address Match (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus Host Header Address received since the last read of TWIHS_SR. |

| Value | Description |
|---|---|
| 1 | An SMBus Host Header Address was received since the last read of TWIHS_SR. |

**Bit 20 – SMBDAM** SMBus Default Address Match (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus Default Address received since the last read of TWIHS_SR. |
| 1 | An SMBus Default Address was received since the last read of TWIHS_SR. |

**Bit 19 – PECERR** PEC Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus PEC error occurred since the last read of TWIHS_SR. |
| 1 | An SMBus PEC error occurred since the last read of TWIHS_SR. |

**Bit 18 – TOUT** Timeout Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus timeout occurred since the last read of TWIHS_SR. |
| 1 | An SMBus timeout occurred since the last read of TWIHS_SR. |

**Bit 16 – MCACK** Master Code Acknowledge (cleared on read)
MACK used in Slave mode:

| Value | Description |
|---|---|
| 0 | No Master Code has been received since the last read of TWIHS_SR. |
| 1 | A Master Code has been received since the last read of TWIHS_SR. |

**Bit 11 – EOSACC** End Of Slave Access (cleared on read)
This bit is used in Slave mode only.

EOSACC behavior can be seen in Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | A slave access is being performing. |
| 1 | The Slave Access is finished. End Of Slave Access is automatically set as soon as SVACC is reset. |

**Bit 10 – SCLWS** Clock Wait State
This bit is used in Slave mode only.

SCLWS behavior can be seen in the figures, Clock Stretching in Read Mode and Clock Stretching in Write Mode.

| Value | Description |
|---|---|
| 0 | The clock is not stretched. |
| 1 | The clock is stretched. TWIHS_THR / TWIHS_RHR buffer is not filled / emptied before the transmission / reception of a new character. |

**Bit 9 – ARBLST** Arbitration Lost (cleared on read)
This bit is used in Master mode only.

| Value | Description |
|---|---|
| 0 | Arbitration won. |
| 1 | Arbitration lost. Another master of the TWIHS bus has won the multimaster arbitration. TXCOMP is set at the same time. |

**Bit 8 – NACK** Not Acknowledged (cleared on read)

- NACK used in Master mode:

0: Each data byte has been correctly received by the far-end side TWIHS slave component.

1: A data or address byte has not been acknowledged by the slave component. Set at the same time as TXCOMP.

- NACK used in Slave Read mode:

0: Each data byte has been correctly received by the master.

1: In Read mode, a data byte has not been acknowledged by the master. When NACK is set, the user must not fill TWIHS_THR even if TXRDY is set, because it means that the master stops the data transfer or re-initiate it.

**Note:** In Slave Write mode, all data are acknowledged by the TWIHS.

**Bit 7 – UNRE** Underrun Error (cleared on read)
This bit is used only if clock stretching is disabled.

| Value | Description |
|---|---|
| 0 | TWIHS_THR has been filled on time. |
| 1 | TWIHS_THR has not been filled on time. |

**Bit 6 – OVRE** Overrun Error (cleared on read)
This bit is used only if clock stretching is disabled.

| Value | Description |
|---|---|
| 0 | TWIHS_RHR has not been loaded while RXRDY was set. |
| 1 | TWIHS_RHR has been loaded while RXRDY was set. Reset by read in TWIHS_SR when TXCOMP is set. |

**Bit 5 – GACC** General Call Access (cleared on read)
This bit is used in Slave mode only.

GACC behavior can be seen in Master Performs a General Call.

| Value | Description |
|---|---|
| 0 | No general call has been detected. |
| 1 | A general call has been detected. After the detection of general call, if need be, the user may acknowledge this access and decode the following bytes and respond according to the value of the bytes. |

**Bit 4 – SVACC** Slave Access
This bit is used in Slave mode only.

SVACC behavior can be seen in Read Access Ordered by a Master, Clock Stretching in Read Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | TWIHS is not addressed. SVACC is automatically cleared after a NACK or a STOP condition is detected. |
| 1 | Indicates that the address decoding sequence has matched (A master has sent SADR). SVACC remains high until a NACK or a STOP condition is detected. |

**Bit 3 – SVREAD** Slave Read
This bit is used in Slave mode only. When SVACC is low (no slave access has been detected) SVREAD is irrelevant.

SVREAD behavior can be seen in Read Access Ordered by a Master, Clock Stretching in Read Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | Indicates that a write access is performed by a master. |
| 1 | Indicates that a read access is performed by a master. |

**Bit 2 – TXRDY** Transmit Holding Register Ready (cleared by writing TWIHS_THR)

- TXRDY used in Master mode:

0: The transmit holding register has not been transferred into the internal shifter. Set to 0 when writing into TWIHS_THR.

1: As soon as a data byte is transferred from TWIHS_THR to internal shifter or if a NACK error is detected, TXRDY is set at the same time as TXCOMP and NACK. TXRDY is also set when MSEN is set (enables TWIHS).

TXRDY behavior in Master mode can be seen in Master Write with One Data Byte, Master Write with Multiple Data Bytes and Master Write with One-Byte Internal Address and Multiple Data Bytes.

- TXRDY used in Slave mode:

0: As soon as data is written in the TWIHS_THR, until this data has been transmitted and acknowledged (ACK or NACK).

1: Indicates that the TWIHS_THR is empty and that data has been transmitted and acknowledged.

If TXRDY is high and if a NACK has been detected, the transmission is stopped. Thus when TRDY = NACK = 1, the user must not fill TWIHS_THR to avoid losing it.

TXRDY behavior in Slave mode can be seen in Read Access Ordered by a Master, Clock Stretching in Read Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

**Bit 1 – RXRDY** Receive Holding Register Ready (cleared by reading TWIHS_RHR)
RXRDY behavior in Master mode can be seen in Master Read with One Data Byte, Master Read with Multiple Data Bytes and Master Read Clock Stretching with Multiple Data Bytes.

RXRDY behavior in Slave mode can be seen in Write Access Ordered by a Master, Clock Stretching in Write Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | No character has been received since the last TWIHS_RHR read operation. |

| Value | Description |
|---|---|
| 1 | A byte has been received in the TWIHS_RHR since the last read. |

**Bit 0 – TXCOMP** Transmission Completed (cleared by writing TWIHS_THR)

- TXCOMP used in Master mode:

0: During the length of the current frame.

1: When both holding register and internal shifter are empty and STOP condition has been sent.

TXCOMP behavior in Master mode can be seen in Master Write with One-Byte Internal Address and Multiple Data Bytes and in Master Read with Multiple Data Bytes.

- TXCOMP used in Slave mode:

0: As soon as a START is detected.

1: After a STOP or a REPEATED START + an address different from SADR is detected.

TXCOMP behavior in Slave mode can be seen in Clock Stretching in Read Mode, Clock Stretching in Write Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

### 45.7.8 TWIHS Status Register (FIFO_ENABLED)

**Name:** TWIHS_SR (FIFO_ENABLED)
**Offset:** 0x20
**Reset:** 0x03000009
**Property:** Read-only

This configuration is relevant only if TWIHS_CR.FIFOEN = ‘1’.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | SDA | SCL |
| Access | | | | | | | R | R |
| Reset | | | | | | | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TXFLOCK | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| Access | R | | R | R | R | R | | R |
| Reset | 0 | | 0 | 0 | 0 | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOSACC | SCLWS | ARBLST | NACK |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | GACC | SVACC | SVREAD | TXRDY | RXRDY | TXCOMP |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |

**Bit 25 – SDA** SDA Line Value

| Value | Description |
|---|---|
| 0 | SDA line sampled value is ‘0’. |
| 1 | SDA line sampled value is ‘1’. |

**Bit 24 – SCL** SCL Line Value

| Value | Description |
|---|---|
| 0 | SCL line sampled value is ‘0’. |
| 1 | SCL line sampled value is ‘1.’ |

**Bit 23 – TXFLOCK** Transmit FIFO Lock

| Value | Description |
|---|---|
| 0 | The Transmit FIFO is not locked. |
| 1 | The Transmit FIFO is locked. |

**Bit 21 – SMBHHM** SMBus Host Header Address Match (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus Host Header Address received since the last read of TWIHS_SR. |

| Value | Description |
|---|---|
| 1 | An SMBus Host Header Address was received since the last read of TWIHS_SR. |

**Bit 20 – SMBDAM** SMBus Default Address Match (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus Default Address received since the last read of TWIHS_SR. |
| 1 | An SMBus Default Address was received since the last read of TWIHS_SR. |

**Bit 19 – PECERR** PEC Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus PEC error occurred since the last read of TWIHS_SR. |
| 1 | An SMBus PEC error occurred since the last read of TWIHS_SR. |

**Bit 18 – TOUT** Timeout Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus timeout occurred since the last read of TWIHS_SR. |
| 1 | An SMBus timeout occurred since the last read of TWIHS_SR. |

**Bit 16 – MCACK** Master Code Acknowledge (cleared on read)

- MACK used in Slave mode:

| Value | Description |
|---|---|
| 0 | No Master Code has been received since the last read of TWIHS_SR. |
| 1 | A Master Code has been received since the last read of TWIHS_SR. |

**Bit 11 – EOSACC** End Of Slave Access (cleared on read)
This bit is used in Slave mode only.

EOSACC behavior can be seen in Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | A slave access is being performing. |
| 1 | The Slave Access is finished. End Of Slave Access is automatically set as soon as SVACC is reset. |

**Bit 10 – SCLWS** Clock Wait State
This bit is used in Slave mode only.

SCLWS behavior can be seen in Clock Stretching in Read Mode and Clock Stretching in Write Mode.

| Value | Description |
|---|---|
| 0 | The clock is not stretched. |
| 1 | The clock is stretched. TWIHS_THR / TWIHS_RHR buffer is not filled / emptied before the transmission / reception of a new character. |

**Bit 9 – ARBLST** Arbitration Lost (cleared on read)
This bit is used in Master mode only.

| Value | Description |
|---|---|
| 0 | Arbitration won. |
| 1 | Arbitration lost. Another master of the TWIHS bus has won the multimaster arbitration. TXCOMP is set at the same time. |

**Bit 8 – NACK** Not Acknowledged (cleared on read)
**Note:** In Slave Write mode, all data are acknowledged by the TWIHS.

- NACK used in Master mode:

0: Each data byte has been correctly received by the far-end side TWIHS slave component.

1: A data or address byte has not been acknowledged by the slave component. Set at the same time as TXCOMP.

- NACK used in Slave Read mode:

0: Each data byte has been correctly received by the master.

1: In Read mode, a data byte has not been acknowledged by the master. When NACK is set, the user must not fill TWIHS_THR even if TXRDY is set, because it means that the master stops the data transfer or re-initiate it.

**Bit 7 – UNRE** Underrun Error (cleared on read)
This bit is used only if clock stretching is disabled.

| Value | Description |
|---|---|
| 0 | TWIHS_THR has been filled on time. |
| 1 | TWIHS_THR has not been filled on time. |

**Bit 6 – OVRE** Overrun Error (cleared on read)
This bit is used only if clock stretching is disabled.

| Value | Description |
|---|---|
| 0 | TWIHS_RHR has not been loaded while RXRDY was set. |
| 1 | TWIHS_RHR has been loaded while RXRDY was set. Reset by read in TWIHS_SR when TXCOMP is set. |

**Bit 5 – GACC** General Call Access (cleared on read)
This bit is used in Slave mode only.

GACC behavior can be seen in Master Performs a General Call.

| Value | Description |
|---|---|
| 0 | No general call has been detected. |
| 1 | A general call has been detected. After the detection of general call, if need be, the user may acknowledge this access and decode the following bytes and respond according to the value of the bytes. |

**Bit 4 – SVACC** Slave Access
This bit is used in Slave mode only.

SVACC behavior can be seen in Read Access Ordered by a Master, Write Access Ordered by a Master, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | TWIHS is not addressed. SVACC is automatically cleared after a NACK or a STOP condition is detected. |
| 1 | Indicates that the address decoding sequence has matched (A master has sent SADR). SVACC remains high until a NACK or a STOP condition is detected. |

**Bit 3 – SVREAD** Slave Read
This bit is used in Slave mode only. When SVACC is low (no slave access has been detected) SVREAD is irrelevant.

SVREAD behavior can be seen in Read Access Ordered by a Master, Write Access Ordered by a Master, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | Indicates that a write access is performed by a master. |
| 1 | Indicates that a read access is performed by a master. |

**Bit 2 – TXRDY** Transmit Holding Register Ready (cleared by writing TWIHS_THR)

- TXRDY used in Master mode:

TXRDY behavior in Master mode can be seen in Master Write with One Data Byte, Master Write with Multiple Data Bytes and Master Write with One-Byte Internal Address and Multiple Data Bytes.

- TXRDY used in Slave mode:

If TXRDY is high and if a NACK has been detected, the transmission is stopped. Thus when TRDY = NACK = 1, the user must not fill TWIHS_THR to avoid losing it.

TXRDY behavior in Slave mode can be seen in Read Access Ordered by a Master, Clock Stretching in Read Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | The transmit holding register has not been transferred into the internal shifter. Set to 0 when writing into TWIHS_THR. |
| 1 | As soon as a data byte is transferred from TWIHS_THR to internal shifter or if a NACK error is detected, TXRDY is set at the same time as TXCOMP and NACK. TXRDY is also set when MSEN is set (enables TWIHS). |
| 0 | As soon as data is written in the TWIHS_THR, until this data has been transmitted and acknowledged (ACK or NACK). |
| 1 | Indicates that the TWIHS_THR is empty and that data has been transmitted and acknowledged. |

**Bit 1 – RXRDY** Receive Holding Register Ready (cleared by reading TWIHS_RHR)
RXRDY behavior in Master mode can be seen in Master Read with One Data Byte, Master Read with Multiple Data Bytes and Master Read Clock Stretching with Multiple Data Bytes.

RXRDY behavior in Slave mode can be seen in Write Access Ordered by a Master, Clock Stretching in Write Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | No character has been received since the last TWIHS_RHR read operation. |

| Value | Description |
|---|---|
| 1 | A byte has been received in the TWIHS_RHR since the last read. |

**Bit 0 – TXCOMP** Transmission Completed (cleared by writing TWIHS_THR)

- TXCOMP used in Master mode:

0: During the length of the current frame.

1: When both holding register and internal shifter are empty and STOP condition has been sent.

TXCOMP behavior in Master mode can be seen in Master Write with One-Byte Internal Address and Multiple Data Bytes and in Master Read with Multiple Data Bytes.

- TXCOMP used in Slave mode:

0: As soon as a START is detected.

1: After a STOP or a REPEATED START + an address different from SADR is detected.

TXCOMP behavior in Slave mode can be seen in Clock Stretching in Read Mode, Clock Stretching in Write Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

### 45.7.9 TWIHS SMBus Timing Register

**Name:** TWIHS_SMBTR
**Offset:** 0x38
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TWIHS Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | THMAX[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TLOWM[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TLOWS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PRESC[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 31:24 – THMAX[7:0]** Clock High Maximum Cycles
Clock cycles in clock high maximum count. Prescaled by PRESC. Used for bus free detection. Used to time THIGH:MAX.

**Bits 23:16 – TLOWM[7:0]** Master Clock Stretch Maximum Cycles

| Value | Description |
|---|---|
| 0 | TLOW:MEXT timeout check disabled. |
| 1-255 | Clock cycles in master maximum clock stretch count. Prescaled by PRESC. Used to time TLOW:MEXT. |

**Bits 15:8 – TLOWS[7:0]** Slave Clock Stretch Maximum Cycles

| Value | Description |
|---|---|
| 0 | TLOW:SEXT timeout check disabled. |
| 1-255 | Clock cycles in slave maximum clock stretch count. Prescaled by PRESC. Used to time TLOW:SEXT. |

**Bits 3:0 – PRESC[3:0]** SMBus Clock Prescaler
Used to specify how to prescale the TLOWS, TLOWM and THMAX counters in SMBTR. Counters are prescaled according to the following formula:

$$f_{\text{Prescaled}} = \frac{f_{\text{peripheral clock}}}{2^{(\text{PRESC} + 1)}}$$

### 45.7.10 TWIHS Alternative Command Register

**Name:** TWIHS_ACR
**Offset:** 0x40
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | NPEC | NDIR |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NDATAL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | PEC | DIR |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATAL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 25 – NPEC** Next PEC Request (SMBus Mode only)

| Value | Description |
|---|---|
| 0 | The next transfer does not use a PEC byte. |
| 1 | The next transfer uses a PEC byte. |

**Bit 24 – NDIR** Next Transfer Direction

| Value | Description |
|---|---|
| 0 | Write direction. |
| 1 | Read direction. |

**Bits 23:16 – NDATAL[7:0]** Next Data Length

| Value | Description |
|---|---|
| 0 | No data to send (see Alternative Command. |
| 1-255 | Number of bytes to send for the next transfer. |

**Bit 9 – PEC** PEC Request (SMBus Mode only)

| Value | Description |
|---|---|
| 0 | The transfer does not use a PEC byte. |
| 1 | The transfer uses a PEC byte. |

**Bit 8 – DIR** Transfer Direction

| Value | Description |
|---|---|
| 0 | Write direction. |
| 1 | Read direction. |

**Bits 7:0 – DATAL[7:0]** Data Length

| Value | Description |
|---|---|
| 0 | No data to send (see Alternative Command). |
| 1-255 | Number of bytes to send during the transfer. |

### 45.7.11 TWIHS Filter Register

**Name:** TWIHS_FILTR
**Offset:** 0x44
**Reset:** 0x00000000
**Property:** Read/Write

TWIHS digital input filtering follows a majority decision based on three samples from SDA/SCL lines at peripheral clock frequency.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | THRES[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PADFCFG | PADFEN | FILT |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 10:8 – THRES[2:0]** Digital Filter Threshold

| Value | Description |
|---|---|
| 0 | No filtering applied on TWIHS inputs. |
| 1-7 | Maximum pulse width of spikes to be suppressed by the input filter, defined in peripheral clock cycles. |

**Bit 2 – PADFCFG** PAD Filter Config
See the electrical characteristics section for filter configuration details.

**Bit 1 – PADFEN** PAD Filter Enable

| Value | Description |
|---|---|
| 0 | PAD analog filter is disabled. |
| 1 | PAD analog filter is enabled. (The analog filter must be enabled if High-speed mode is enabled.) |

**Bit 0 – FILT** RX Digital Filter

| Value | Description |
|---|---|
| 0 | No filtering applied on TWIHS inputs. |
| 1 | TWIHS input filtering is active (only in Standard and Fast modes) |

### 45.7.12 TWIHS Interrupt Enable Register

**Name:** TWIHS_IER
**Offset:** 0x24
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| Access | | | W | W | W | W | | W |
| Reset | | | – | – | – | – | | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOSACC | SCL_WS | ARBLST | NACK |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| Access | W | W | W | W | | W | W | W |
| Reset | – | – | – | – | | – | – | – |

**Bit 21 – SMBHHM** SMBus Host Header Address Match Interrupt Enable

**Bit 20 – SMBDAM** SMBus Default Address Match Interrupt Enable

**Bit 19 – PECERR** PEC Error Interrupt Enable

**Bit 18 – TOUT** Timeout Error Interrupt Enable

**Bit 16 – MCACK** Master Code Acknowledge Interrupt Enable

**Bit 11 – EOSACC** End Of Slave Access Interrupt Enable

**Bit 10 – SCL_WS** Clock Wait State Interrupt Enable

**Bit 9 – ARBLST** Arbitration Lost Interrupt Enable

**Bit 8 – NACK** Not Acknowledge Interrupt Enable

**Bit 7 – UNRE** Underrun Error Interrupt Enable

**Bit 6 – OVRE** Overrun Error Interrupt Enable

**Bit 5 – GACC** General Call Access Interrupt Enable

**Bit 4 – SVACC** Slave Access Interrupt Enable

**Bit 2 – TXRDY** Transmit Holding Register Ready Interrupt Enable

**Bit 1 – RXRDY** Receive Holding Register Ready Interrupt Enable

**Bit 0 – TXCOMP** Transmission Completed Interrupt Enable

### 45.7.13 TWIHS Interrupt Disable Register

**Name:** TWIHS_IDR
**Offset:** 0x28
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| Access | | | W | W | W | W | | W |
| Reset | | | – | – | – | – | | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOSACC | SCL_WS | ARBLST | NACK |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| Access | W | W | W | W | | W | W | W |
| Reset | – | – | – | – | | – | – | – |

**Bit 21 – SMBHHM** SMBus Host Header Address Match Interrupt Disable

**Bit 20 – SMBDAM** SMBus Default Address Match Interrupt Disable

**Bit 19 – PECERR** PEC Error Interrupt Disable

**Bit 18 – TOUT** Timeout Error Interrupt Disable

**Bit 16 – MCACK** Master Code Acknowledge Interrupt Disable

**Bit 11 – EOSACC** End Of Slave Access Interrupt Disable

**Bit 10 – SCL_WS** Clock Wait State Interrupt Disable

**Bit 9 – ARBLST** Arbitration Lost Interrupt Disable

**Bit 8 – NACK** Not Acknowledge Interrupt Disable

**Bit 7 – UNRE** Underrun Error Interrupt Disable

**Bit 6 – OVRE** Overrun Error Interrupt Disable

**Bit 5 – GACC** General Call Access Interrupt Disable

**Bit 4 – SVACC** Slave Access Interrupt Disable

**Bit 2 – TXRDY** Transmit Holding Register Ready Interrupt Disable

**Bit 1 – RXRDY** Receive Holding Register Ready Interrupt Disable

**Bit 0 – TXCOMP** Transmission Completed Interrupt Disable

### 45.7.14 TWIHS Interrupt Mask Register

**Name:** TWIHS_IMR
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| Access | | | R | R | R | R | | R |
| Reset | | | 0 | 0 | 0 | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOSACC | SCL_WS | ARBLST | NACK |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| Access | R | R | R | R | | R | R | R |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bit 21 – SMBHHM** SMBus Host Header Address Match Interrupt Mask

**Bit 20 – SMBDAM** SMBus Default Address Match Interrupt Mask

**Bit 19 – PECERR** PEC Error Interrupt Mask

**Bit 18 – TOUT** Timeout Error Interrupt Mask

**Bit 16 – MCACK** Master Code Acknowledge Interrupt Mask

**Bit 11 – EOSACC** End Of Slave Access Interrupt Mask

**Bit 10 – SCL_WS** Clock Wait State Interrupt Mask

**Bit 9 – ARBLST** Arbitration Lost Interrupt Mask

**Bit 8 – NACK** Not Acknowledge Interrupt Mask

**Bit 7 – UNRE** Underrun Error Interrupt Mask

**Bit 6 – OVRE** Overrun Error Interrupt Mask

**Bit 5 – GACC** General Call Access Interrupt Mask

**Bit 4 – SVACC** Slave Access Interrupt Mask

**Bit 2 – TXRDY** Transmit Holding Register Ready Interrupt Mask

**Bit 1 – RXRDY** Receive Holding Register Ready Interrupt Mask

**Bit 0 – TXCOMP** Transmission Completed Interrupt Mask

### 45.7.15 TWIHS Receive Holding Register

**Name:** TWIHS_RHR
**Offset:** 0x30
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXDATA[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – RXDATA[7:0]** Master or Slave Receive Holding Data

### 45.7.16 TWIHS Receive Holding Register (FIFO Enabled)

**Name:** TWIHS_RHR (FIFO_ENABLED)
**Offset:** 0x30
**Reset:** 0x00000000
**Property:** Read-only

If FIFO is enabled (FIFOEN bit in TWIHS_CR), refer to 45.6.6.8 Multiple Data Mode for details.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RXDATA3[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RXDATA2[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RXDATA1[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXDATA0[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – RXDATA3[7:0]** Master or Slave Receive Holding Data 3

**Bits 23:16 – RXDATA2[7:0]** Master or Slave Receive Holding Data 2

**Bits 15:8 – RXDATA1[7:0]** Master or Slave Receive Holding Data 1

**Bits 7:0 – RXDATA0[7:0]** Master or Slave Receive Holding Data 0

### 45.7.17 TWIHS SleepWalking Matching Register

**Name:** TWIHS_SWMR
**Offset:** 0x4C
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TWIHS Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATAM[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | SADR3[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | SADR2[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | SADR1[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – DATAM[7:0]** Data Match
The TWIHS module extends the SleepWalking matching process to the first received data, comparing it with DATAM if DATAMEN bit is enabled.

**Bits 22:16 – SADR3[6:0]** Slave Address 3
Slave address 3. The TWIHS module matches on this additional address if SADR3EN bit is enabled.

**Bits 14:8 – SADR2[6:0]** Slave Address 2
Slave address 2. The TWIHS module matches on this additional address if SADR2EN bit is enabled.

**Bits 6:0 – SADR1[6:0]** Slave Address 1
Slave address 1. The TWIHS module matches on this additional address if SADR1EN bit is enabled.

### 45.7.18 TWIHS Transmit Holding Register

**Name:** TWIHS_THR
**Offset:** 0x34
**Reset:** 0x00000000
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXDATA[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – TXDATA[7:0]** Master or Slave Transmit Holding Data

### 45.7.19 TWIHS Transmit Holding Register (FIFO Enabled)

**Name:** TWIHS_THR (FIFO_ENABLED)
**Offset:** 0x34
**Reset:** 0x00000000
**Property:** Write-only

If FIFO is enabled (FIFOEN bit in TWIHS_CR), refer to 45.6.6.8 Multiple Data Mode for details.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TXDATA3[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TXDATA2[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TXDATA1[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXDATA0[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – TXDATA3[7:0]** Master or Slave Transmit Holding Data 3

**Bits 23:16 – TXDATA2[7:0]** Master or Slave Transmit Holding Data 2

**Bits 15:8 – TXDATA1[7:0]** Master or Slave Transmit Holding Data 1

**Bits 7:0 – TXDATA0[7:0]** Master or Slave Transmit Holding Data 02

### 45.7.20 TWIHS FIFO Mode Register

**Name:** TWIHS_FMR
**Offset:** 0x50
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | RXFTHRES[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | TXFTHRES[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | RXRDYM[1:0] | | | | TXRDYM[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bits 29:24 – RXFTHRES[5:0]** Receive FIFO Threshold

| Value | Description |
|---|---|
| 0–16 | Defines the Receive FIFO threshold value (number of data). RXFTH flag in TWIHS_FSR will be set when Receive FIFO goes from “below” threshold state to “equal or above” threshold state. |

**Bits 21:16 – TXFTHRES[5:0]** Transmit FIFO Threshold

| Value | Description |
|---|---|
| 0–16 | Defines the Transmit FIFO threshold value (number of data). TXFTH flag in TWIHS_FSR will be set when Transmit FIFO goes from “above” threshold state to “equal or below” threshold state. |

**Bits 5:4 – RXRDYM[1:0]** Receiver Ready Mode
If FIFOs are enabled, the RXRDY flag (in TWIHS_SR) behaves as follows.

| Value | Name | Description |
|---|---|---|
| 0 | ONE_DATA | RXRDY will be at level ‘1’ when at least one unread data is in the Receive FIFO |
| 1 | TWO_DATA | RXRDY will be at level ‘1’ when at least two unread data are in the Receive FIFO |

| Value | Name | Description |
|---|---|---|
| 2 | FOUR_DATA | RXRDY will be at level ‘1’ when at least four unread data are in the Receive FIFO |

**Bits 1:0 – TXRDYM[1:0]** Transmitter Ready Mode
If FIFOs are enabled, the TXRDY flag (in TWIHS_SR) behaves as follows.

| Value | Name | Description |
|---|---|---|
| 0 | ONE_DATA | TXRDY will be at level ‘1’ when at least one data can be written in the Transmit FIFO |
| 1 | TWO_DATA | TXRDY will be at level ‘1’ when at least two data can be written in the Transmit FIFO |
| 2 | FOUR_DATA | TXRDY will be at level ‘1’ when at least four data can be written in the Transmit FIFO |

### 45.7.21 TWIHS FIFO Level Register

**Name:** TWIHS_FLR
**Offset:** 0x54
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | RXFL[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TXFL[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 21:16 – RXFL[5:0]** Receive FIFO Level

| Value | Description |
|---|---|
| 0 | There is no unread data in the Receive FIFO |
| 1-16 | Indicates the number of unread DATA in the Receive FIFO |

**Bits 5:0 – TXFL[5:0]** Transmit FIFO Level

| Value | Description |
|---|---|
| 0 | There is no data in the Transmit FIFO |
| 1-16 | Indicates the number of DATA in the Transmit FIFO |

### 45.7.22 TWIHS FIFO Status Register

**Name:** TWIHS_FSR
**Offset:** 0x60
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – RXFPTEF** Receive FIFO Pointer Error Flag
See 45.6.6.10 FIFO Pointer Error for details.

| Value | Description |
|---|---|
| 0 | No Receive FIFO pointer occurred |
| 1 | Receive FIFO pointer error occurred. Receiver must be reset |

**Bit 6 – TXFPTEF** Transmit FIFO Pointer Error Flag
See 45.6.6.10 FIFO Pointer Error for details.

| Value | Description |
|---|---|
| 0 | No Transmit FIFO pointer occurred |
| 1 | Transmit FIFO pointer error occurred. Transceiver must be reset |

**Bit 5 – RXFTHF** Receive FIFO Threshold Flag

| Value | Description |
|---|---|
| 0 | Number of unread DATA in Receive FIFO is below RXFTHRES threshold. |
| 1 | Number of unread DATA in Receive FIFO has reached RXFTHRES threshold since the last read of TWIHS_FSR. |

**Bit 4 – RXFFF** Receive FIFO Full Flag

| Value | Description |
|---|---|
| 0 | Receive FIFO is not empty. |
| 1 | Receive FIFO has been filled since the last read of TWIHS_FSR. |

**Bit 3 – RXFEF** Receive FIFO Empty Flag

| Value | Description |
|---|---|
| 0 | Receive FIFO is not empty. |
| 1 | Receive FIFO has been emptied since the last read of TWIHS_FSR. |

**Bit 2 – TXFTHF** Transmit FIFO Threshold Flag (cleared on read)

| Value | Description |
|---|---|
| 0 | Number of DATA in Transmit FIFO is above TXFTHRES threshold. |
| 1 | Number of DATA in Transmit FIFO has reached TXFTHRES threshold since the last read of TWIHS_FSR. |

**Bit 1 – TXFFF** Transmit FIFO Full Flag (cleared on read)

| Value | Description |
|---|---|
| 0 | Transmit FIFO is not full. |
| 1 | Transmit FIFO has been filled since the last read of TWIHS_FSR. |

**Bit 0 – TXFEF** Transmit FIFO Empty Flag (cleared on read)

| Value | Description |
|---|---|
| 0 | Transmit FIFO is not empty. |
| 1 | Transmit FIFO has been emptied since the last read of TWIHS_FSR. |

### 45.7.23 TWIHS FIFO Interrupt Enable Register

**Name:** TWIHS_FIER
**Offset:** 0x64
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 7 – RXFPTEF** RXFPTEF Interrupt Enable

**Bit 6 – TXFPTEF** TXFPTEF Interrupt Enable

**Bit 5 – RXFTHF** RXFTHF Interrupt Enable

**Bit 4 – RXFFF** RXFFF Interrupt Enable

**Bit 3 – RXFEF** RXFEF Interrupt Enable

**Bit 2 – TXFTHF** TXFTHF Interrupt Enable

**Bit 1 – TXFFF** TXFFF Interrupt Enable

**Bit 0 – TXFEF** TXFEF Interrupt Enable

### 45.7.24 TWIHS FIFO Interrupt Disable Register

**Name:** TWIHS_FIDR
**Offset:** 0x68
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 7 – RXFPTEF** RXFPTEF Interrupt Disable

**Bit 6 – TXFPTEF** TXFPTEF Interrupt Disable

**Bit 5 – RXFTHF** RXFTHF Interrupt Disable

**Bit 4 – RXFFF** RXFFF Interrupt Disable

**Bit 3 – RXFEF** RXFEF Interrupt Disable

**Bit 2 – TXFTHF** TXFTHF Interrupt Disable

**Bit 1 – TXFFF** TXFFF Interrupt Disable

**Bit 0 – TXFEF** TXFEF Interrupt Disable

### 45.7.25 TWIHS FIFO Interrupt Mask Register

**Name:** TWIHS_FIMR
**Offset:** 0x6C
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – RXFPTEF** RXFPTEF Interrupt Mask

**Bit 6 – TXFPTEF** TXFPTEF Interrupt Mask

**Bit 5 – RXFTHF** RXFTHF Interrupt Mask

**Bit 4 – RXFFF** RXFFF Interrupt Mask

**Bit 3 – RXFEF** RXFEF Interrupt Mask

**Bit 2 – TXFTHF** TXFTHF Interrupt Mask

**Bit 1 – TXFFF** TXFFF Interrupt Mask

**Bit 0 – TXFEF** TXFEF Interrupt Mask

### 45.7.26 TWIHS Write Protection Mode Register

**Name:** TWIHS_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x545749 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See Register Write Protection for the list of registers that can be write-protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x545749 (“TWI” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x545749 (“TWI” in ASCII). |

### 45.7.27 TWIHS Write Protection Status Register

**Name:** TWIHS_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPVSRC[23:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of the TWIHS_WPSR. |
| 1 | A write protection violation has occurred since the last read of the TWIHS_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 46. Flexible Serial Communication Controller (FLEXCOM)

## 46.1 Description

The Flexible Serial Communication Controller (FLEXCOM) offers several serial communication protocols that are managed by the three submodules USART, SPI, and TWI.

The Universal Synchronous Asynchronous Receiver Transceiver (USART) provides one full-duplex universal synchronous asynchronous serial link. Data frame format is widely programmable (data length, parity, number of stop bits) to support a maximum of standards. The receiver implements parity error, framing error and overrun error detection. The receiver timeout enables handling variable-length frames and the transmitter timeguard facilitates communications with slow remote devices. Multidrop communications are also supported through address bit handling in reception and transmission.

The USART features three test modes: Remote Loopback, Local Loopback and Automatic Echo.

The USART supports specific operating modes providing interfaces on RS485, LIN, and SPI, with ISO7816 T = 0 or T = 1 smart card slots, and infrared transceivers. The hardware handshaking feature enables an out-of-band flow control by automatic management of the pins RTS and CTS.

The USART supports the connection to the DMA Controller, which enables data transfers to the transmitter and from the receiver. The DMAC provides chained buffer management without any intervention of the processor.

The Serial Peripheral Interface (SPI) circuit is a synchronous serial data link that provides communication with external devices in Master or Slave mode. It also enables communication between processors if an external processor is connected to the system.

The Serial Peripheral Interface is essentially a shift register that serially transmits data bits to other SPIs. During a data transfer, one SPI system acts as the “master” which controls the data flow, while the other devices act as “slaves” which have data shifted into and out by the master. Different CPUs can take turn being masters (multiple master protocol, contrary to single master protocol where one CPU is always the master while all of the others are always slaves). One master can simultaneously shift data into multiple slaves. However, only one slave can drive its output to write data back to the master at any given time.

A slave device is selected when the master asserts its NSS signal. If multiple slave devices exist, the master generates a separate slave select signal for each slave (NPCS).

The SPI system consists of two data lines and two control lines:

- Master Out Slave In (MOSI)—This data line supplies the output data from the master shifted into the input(s) of the slave(s).
- Master In Slave Out (MISO)—This data line supplies the output data from a slave to the input of the master. There may be no more than one slave transmitting data during any particular transfer.
- Serial Clock (SPCK)—This control line is driven by the master and regulates the flow of the data bits. The master can transmit data at a variety of baud rates; there is one SPCK pulse for each bit that is transmitted.
- Slave Select (NSS)—This control line allows slaves to be turned on and off by hardware.

The Two-wire Interface (TWI) interconnects components on a unique two-wire bus, made up of one clock line and one data line with speeds of up to 400 kbits per second in Fast mode and up to 3.4 Mbits per second in High-speed Slave mode only, based on a byte-oriented transfer format. It can be used with any Two-wire Interface bus Serial EEPROM and $I^2C$-compatible devices, such as a Real-Time Clock (RTC),

Dot Matrix/Graphic LCD Controller and temperature sensor. The TWI is programmable as a master or a slave with sequential or single-byte access. Multiple master capability is supported.

Arbitration of the bus is performed internally and puts the TWI in Slave mode automatically if the bus arbitration is lost.

A configurable baud rate generator permits the output data rate to be adapted to a wide range of core clock frequencies.

The following table lists the compatibility level of any Two-wire Interface in Master mode and a full $I^2C$ compatible device.

**Table 46-1. TWI Compatibility with $I^2C$ Standard**

| $I^2C$ Standard | TWI |
|---|---|
| Standard Mode Speed (100 kHz) | Supported |
| Fast Mode Speed (400 kHz) | Supported |
| High-speed Mode (Slave only, 3.4 MHz) | Supported |
| 7- or 10-bit(1) Slave Addressing | Supported |
| Repeated Start (Sr) Condition | Supported |
| ACK and NACK Management | Supported |
| Input Filtering | Supported |
| Slope Control | Not Supported |
| Clock Stretching | Supported |
| Multi Master Capability | Supported |

Notes: 1. 10-bit support in Master mode only

## 46.2 Embedded Characteristics

### 46.2.1 USART/UART Characteristics

- 32-byte Transmit and Receive FIFOs
- Programmable Baud Rate Generator
- Baud Rate can be Independent of the Processor/Peripheral Clock
- Comparison Function on Received Character
- 5-bit to 9-bit Full-duplex Synchronous or Asynchronous Serial Communications
  - 1, 1.5 or 2 Stop Bits in Asynchronous Mode or 1 or 2 Stop Bits in Synchronous Mode
  - Parity Generation and Error Detection
  - Framing Error Detection, Overrun Error Detection
  - Digital Filter on Receive Line
  - MSB- or LSB-first
  - Optional Break Generation and Detection
  - By 8 or by 16 Oversampling Receiver Frequency
  - Optional Hardware Handshaking RTS-CTS

  - Receiver Timeout and Transmitter Timeguard
  - Optional Multidrop Mode with Address Generation and Detection
- RS485 with Driver Control Signal
- ISO7816, T = 0 or T = 1 Protocols for Interfacing with Smart Cards
  - NACK Handling, Error Counter with Repetition and Iteration Limit
- IrDA Modulation and Demodulation
  - Communication at up to 115.2 kbit/s
- SPI Mode
  - MASTER or Slave
  - Serial Clock Programmable Phase and Polarity
  - SPI Serial Clock (SCK) Frequency up to $f_{peripheral\ clock}/6$
- LIN Mode
  - Compliant with LIN 1.3 and LIN 2.0 specifications
  - Master or Slave
  - Processing of Frames with Up to 256 Data Bytes
  - Response Data Length can be Configurable or Defined Automatically by the Identifier
  - Self-synchronization in Slave Node Configuration
  - Automatic Processing and Verification of the “Synch Break” and the “Synch Field”
  - “Synch Break” Detection Even When Partially Superimposed with a Data Byte
  - Automatic Identifier Parity Calculation/Sending and Verification
  - Parity Sending and Verification Can be Disabled
  - Automatic Checksum Calculation/sending and Verification
  - Checksum Sending and Verification Can be Disabled
  - Support Both “Classic” and “Enhanced” Checksum Types
  - Full LIN Error Checking and Reporting
  - Frame Slot Mode: Master Allocates Slots to the Scheduled Frames Automatically
  - Generation of the Wakeup Signal
- Test Modes
  - Remote Loopback, Local Loopback, Automatic Echo
- Supports Connection of:
  - Two DMA Controller (DMAC) Channels
  - Offers Buffer Transfer without Processor Intervention
- Register Write Protection

### 46.2.2 SPI Characteristics

- 32-byte Transmit and Receive FIFOs
- Master or Slave Serial Peripheral Bus Interface
  - 8-bit to 16-bit programmable data length per chip select
  - Programmable phase and polarity per chip select
  - Programmable transfer delay between consecutive transfers and delay before SPI clock per chip select
  - Programmable delay between chip selects
- Selectable Mode Fault Detection

- Master Mode Can Drive SPCK up to Peripheral Clock
- Master Mode Bit Rate Can Be Independent of the Processor/Peripheral Clock
- Slave Mode Operates on SPCK, Asynchronously with Core and Bus Clock
- Two Chip Selects with External Decoder Support Allow Communication with up to 3 Peripherals
- Communication with Serial External Devices Supported
  - Serial memories, such as DataFlash and 3-wire EEPROMs
  - Serial peripherals, such as ADCs, DACs, LCD controllers, CAN controllers and sensors
  - External coprocessors
- Connection to DMA Channel Capabilities, Optimizing Data Transfers
  - One channel for the receiver
  - One channel for the transmitter
- Register Write Protection

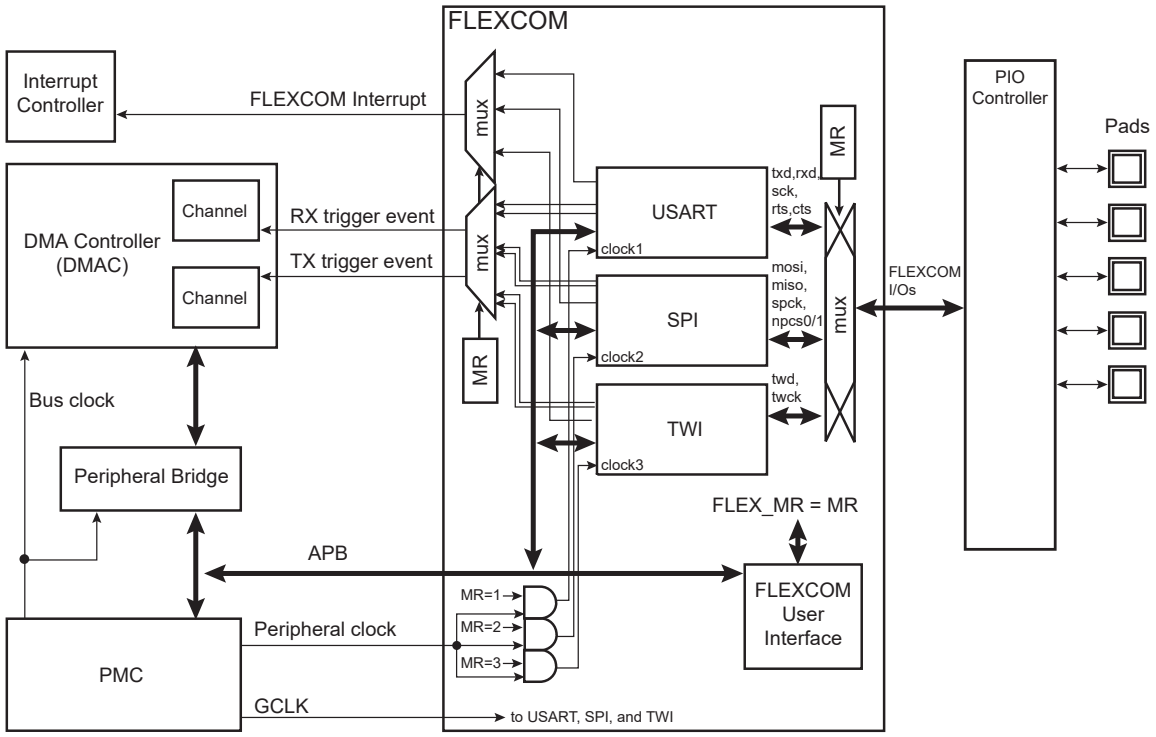
### 46.2.3 TWI/SMBus Characteristics

- 16-byte Transmit and Receive FIFOs
- Bit Rate: Up to 400 kbit/s in Fast Mode and 3.4 Mbit/s in High-Speed Mode (Slave Only)
- Bit Rate can be Independent of the Processor/Peripheral Clock
- SMBus Support
- Compatible with Two-wire Interface Serial Memory and $I^2C$ Compatible Devices(1)
- Master and Multi-Master Operation (Standard and Fast Mode Only)
- Slave Mode Operation (Standard, Fast and High-Speed Mode)
- One, Two or Three Bytes for Slave Address
- Sequential Read/Write Operations
- General Call Supported in Slave Mode
- Connection to DMA Controller Channels Optimizes Data Transfers
  - One Channel for the Receiver
  - One Channel for the Transmitter
- Register Write Protection

Note: 1. See table TWI Compatibility with $I^2C$ Standard for further details.

## 46.3 Block Diagram

**Figure 46-1. FLEXCOM Block Diagram**




## 46.4 I/O Lines Description

**Table 46-2. I/O Lines Description**

| Name | Description | | | Type |
|---|---|---|---|---|
| | USART/UART | SPI | TWI | |
| FLEXCOM_IO0 | TXD | MOSI | TWD | I/O |
| FLEXCOM_IO1 | RXD | MISO | TWCK | I/O |
| FLEXCOM_IO2 | SCK | SPCK | – | I/O |
| FLEXCOM_IO3 | CTS | NPCS0/NSS | – | I/O |
| FLEXCOM_IO4 | RTS | NPCS1 | – | O |

## 46.5 Product Dependencies

### 46.5.1 I/O Lines

The pins used for interfacing the FLEXCOM are multiplexed with the PIO lines. The programmer must first program the PIO controller to assign the desired FLEXCOM pins to their peripheral function. If I/O lines of the FLEXCOM are not used by the application, they can be used for other purposes by the PIO Controller.

### 46.5.2 Power Management

The peripheral clock is not continuously provided to the FLEXCOM. The programmer must first enable the FLEXCOM Clock in the Power Management Controller (PMC) before using the USART or SPI or TWI.

In SleepWalking mode (asynchronous partial wakeup), the PMC must be configured to enable SleepWalking for the FLEXCOM in the Sleepwalking Enable Register (PMC_SLPWK_ER). The peripheral clock can be automatically provided to the FLEXCOM, depending on the instructions (requests) provided by the FLEXCOM to the PMC.

### 46.5.3 Interrupt Sources

The FLEXCOM interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the FLEXCOM interrupt requires the Interrupt Controller to be programmed first.

## 46.6 Register Accesses

Register accesses support 8-bit, 16-bit and 32-bit accesses, which means that only an 8-bit part of a 32-bit register can be written in one access, for instance. For this the access must be done with the right size at the right address.

8-bit, 16-bit and 32-bit accesses are supported for register accesses. However, a field in a register cannot be partially written (e.g., if a field is bigger than 8 bits, the whole field must be written).

This feature helps avoiding a read-modify-write process if only a small part of the register is to be modified.

## 46.7 USART Functional Description

### 46.7.1 Baud Rate Generator

The baud rate generator provides the bit period clock named “baud rate clock” to both the receiver and the transmitter.

Configuring the USCLKS field in FLEX_US_MR selects the baud rate generator clock from one of the following sources:

- the peripheral clock
- a division of the peripheral clock, the divider being product dependent, but generally set to 8
- a fully programmable generic clock (GCLK) provided by PMC and independent of processor/peripheral clock
- the external clock, available on the SCK pin

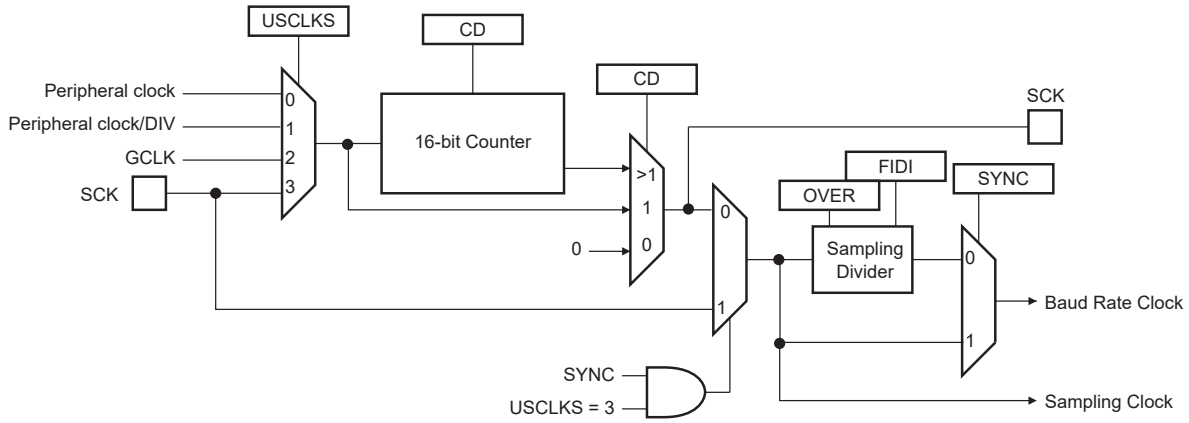
The baud rate generator is based upon a 16-bit divider, which is programmed with the CD field of the Baud Rate Generator Register (FLEX_US_BRGR). If a zero is written to CD, the baud rate generator does not generate any clock. If a one is written to CD, the divider is bypassed and becomes inactive.

If the external SCK clock is selected, the duration of the low and high levels of the signal provided on the SCK pin must be longer than a peripheral clock period. The frequency of the signal provided on SCK must be at least three times lower than peripheral clock in USART mode (field USART_MODE differs from 0xE or 0xF) or six times lower in SPI mode (field USART_MODE equals 0xE or 0xF).

If GCLK is selected, the baud rate is independent of the processor/peripheral clock and thus processor/peripheral clock frequency can be changed without affecting the USART transfer. The GCLK frequency must be at least three times lower than peripheral clock frequency.

If GCLK is selected (USCLKS = 2) and the SCK pin is driven (CLKO = 1), the CD field must be greater than 1.

**Figure 46-2. Baud Rate Generator**




### 46.7.1.1 Baud Rate in Asynchronous Mode

If the USART is programmed to operate in Asynchronous mode, the selected clock is first divided by CD, which is field-programmed in FLEX_US_BRGR. The resulting clock is provided to the receiver as a sampling clock and then divided by 16 or 8, depending on the programming of FLEX_US_MR.OVER.

If OVER is set, the receiver sampling is eight times higher than the baud rate clock. If OVER is cleared, the sampling is performed at 16 times the baud rate clock.

The baud rate is calculated as per the following formula:

$$\text{Baud rate} = \frac{\text{Selected Clock}}{(8(2 - \text{OVER})\text{CD})}$$

This gives a maximum baud rate of peripheral clock divided by 8, assuming that peripheral clock is the highest possible clock and that the OVER bit is set.

#### 46.7.1.1.1 Baud Rate Calculation Example

The following table shows calculations of CD to obtain a baud rate at 38,400 bit/s for different source clock frequencies. It also shows the actual resulting baud rate and the error.

**Table 46-3. Baud Rate Example (OVER = 0)**

| Source Clock (MHz) | Expected Baud Rate (bit/s) | Calculation Result | CD | Actual Baud Rate (bit/s) | Error |
|---|---|---|---|---|---|
| 3,686,400 | 38,400 | 6.00 | 6 | 38,400.00 | 0.00% |
| 4,915,200 | 38,400 | 8.00 | 8 | 38,400.00 | 0.00% |
| 5,000,000 | 38,400 | 8.14 | 8 | 39,062.50 | 1.70% |
| 7,372,800 | 38,400 | 12.00 | 12 | 38,400.00 | 0.00% |
| 8,000,000 | 38,400 | 13.02 | 13 | 38,461.54 | 0.16% |
| 12,000,000 | 38,400 | 19.53 | 20 | 37,500.00 | 2.40% |
| 12,288,000 | 38,400 | 20.00 | 20 | 38,400.00 | 0.00% |
| 14,318,180 | 38,400 | 23.30 | 23 | 38,908.10 | 1.31% |

.....continued

| Source Clock (MHz) | Expected Baud Rate (bit/s) | Calculation Result | CD | Actual Baud Rate (bit/s) | Error |
|---|---|---|---|---|---|
| 14,745,600 | 38,400 | 24.00 | 24 | 38,400.00 | 0.00% |
| 18,432,000 | 38,400 | 30.00 | 30 | 38,400.00 | 0.00% |
| 24,000,000 | 38,400 | 39.06 | 39 | 38,461.54 | 0.16% |
| 24,576,000 | 38,400 | 40.00 | 40 | 38,400.00 | 0.00% |
| 25,000,000 | 38,400 | 40.69 | 40 | 38,109.76 | 0.76% |
| 32,000,000 | 38,400 | 52.08 | 52 | 38,461.54 | 0.16% |
| 32,768,000 | 38,400 | 53.33 | 53 | 38,641.51 | 0.63% |
| 33,000,000 | 38,400 | 53.71 | 54 | 38,194.44 | 0.54% |
| 40,000,000 | 38,400 | 65.10 | 65 | 38,461.54 | 0.16% |
| 50,000,000 | 38,400 | 81.38 | 81 | 38,580.25 | 0.47% |

The baud rate is calculated with the following formula:

Baud rate = MCK / CD × 16

The baud rate error is calculated with the following formula. It is not recommended to work with an error higher than 5%.

$$\text{Error} = 1 - \left(\frac{\text{Expected Baud Rate}}{\text{Actual Baud Rate}}\right)$$

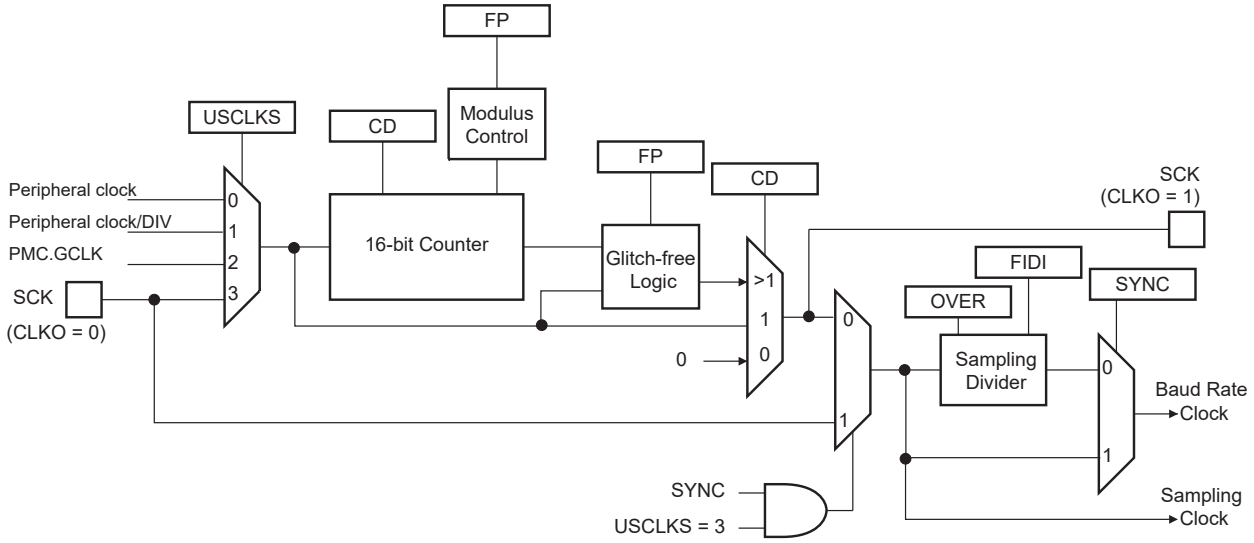
#### 46.7.1.2 Fractional Baud Rate in Asynchronous Mode

The baud rate generator previously defined is subject to the following limitation: the output frequency changes by only integer multiples of the reference frequency. An approach to this problem is to integrate a fractional N clock generator that has a high resolution. The generator architecture is modified to obtain baud rate changes by a fraction of the reference source clock. This fractional part is programmed with the FP field in FLEX_US_BRGR. If FP is not 0, the fractional part is activated. The resolution is one eighth of the clock divider. The fractional baud rate is calculated using the following formula:

$$\text{Baud rate} = \frac{\text{Selected Clock}}{\left(8(2 - \text{OVER})\left(\text{CD} + \frac{\text{FP}}{8}\right)\right)}$$

The modified architecture is presented in the following figure.

**Figure 46-3. Fractional Baud Rate Generator**




**WARNING** When the value of field FP is greater than 0, the SCK (oversampling clock) generates nonconstant duty cycles. The SCK high duration is increased by “selected clock” period from time to time. The duty cycle depends on the value of the CD field.

#### 46.7.1.3 Baud Rate in Synchronous Mode or SPI Mode

If the USART is programmed to operate in Synchronous mode, the selected clock is simply divided by the CD field in FLEX_US_BRGR:

$$\text{Baud rate} = \frac{\text{Selected Clock}}{\text{CD}}$$

In Synchronous mode, if the external clock is selected (USCLKS = 3) and CLKO = 0 (Slave mode), the clock is provided directly by the signal on the USART SCK pin. No division is active. The value written in FLEX_US_BRGR has no effect. The external clock frequency must be at least three times lower than the system clock. In Synchronous mode master (USCLKS = 0 or 1, CLKO = 1), the receive part limits the SCK maximum frequency to $f_{\text{peripheral clock}}/3$ in USART mode, or $f_{\text{peripheral clock}}/6$ in SPI mode.

When either the external clock SCK or the internal clock divided (peripheral clock/DIV or GCLK) is selected, the value programmed in CD must be even if the user has to ensure a 50:50 mark/space ratio on the SCK pin. If the peripheral clock is selected, the baud rate generator ensures a 50:50 duty cycle on the SCK pin, even if the value programmed in CD is odd.

#### 46.7.1.4 Baud Rate in ISO 7816 Mode

The ISO7816 specification defines the bit rate with the following formula:

$$B = \frac{\text{Di}}{\text{Fi}} \times f$$

where:

- B is the bit rate
- Di is the bit rate adjustment factor
- Fi is the clock frequency division factor

- f is the ISO7816 clock frequency (Hz)

Di is a binary value encoded on a 4-bit field, named DI, as represented in the following table.

**Table 46-4. Binary and Decimal Values for Di**

| DI field | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 1000 | 1001 |
|---|---|---|---|---|---|---|---|---|
| Di (decimal) | 1 | 2 | 4 | 8 | 16 | 32 | 12 | 20 |

Fi is a binary value encoded on a 4-bit field, named FI, as represented in the following table.

**Table 46-5. Binary and Decimal Values for Fi**

| FI field | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 1001 | 1010 | 1011 | 1100 | 1101 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Fi (decimal) | 372 | 372 | 558 | 744 | 1116 | 1488 | 1860 | 512 | 768 | 1024 | 1536 | 2048 |

The following table shows the resulting Fi/Di Ratio, which is the ratio between the ISO7816 clock and the baud rate clock.

**Table 46-6. Possible Values for the Fi/Di Ratio**

| Fi/Di | 372 | 558 | 744 | 1116 | 1488 | 1806 | 512 | 768 | 1024 | 1536 | 2048 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 372 | 558 | 744 | 1116 | 1488 | 1860 | 512 | 768 | 1024 | 1536 | 2048 |
| 2 | 186 | 279 | 372 | 558 | 744 | 930 | 256 | 384 | 512 | 768 | 1024 |
| 4 | 93 | 139.5 | 186 | 279 | 372 | 465 | 128 | 192 | 256 | 384 | 512 |
| 8 | 46.5 | 69.75 | 93 | 139.5 | 186 | 232.5 | 64 | 96 | 128 | 192 | 256 |
| 16 | 23.25 | 34.87 | 46.5 | 69.75 | 93 | 116.2 | 32 | 48 | 64 | 96 | 128 |
| 32 | 11.62 | 17.43 | 23.25 | 34.87 | 46.5 | 58.13 | 16 | 24 | 32 | 48 | 64 |
| 12 | 31 | 46.5 | 62 | 93 | 124 | 155 | 42.66 | 64 | 85.33 | 128 | 170.6 |
| 20 | 18.6 | 27.9 | 37.2 | 55.8 | 74.4 | 93 | 25.6 | 38.4 | 51.2 | 76.8 | 102.4 |

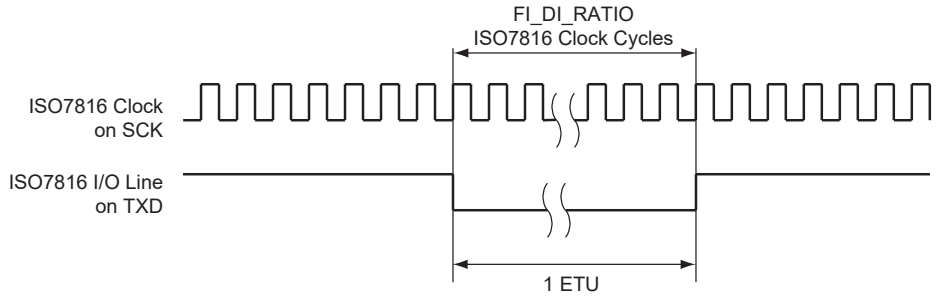
If the USART is configured in ISO7816 mode, the clock selected by the USCLKS field in FLEX_US_MR is first divided by the value programmed in field CD field in FLEX_US_BRGR. The resulting clock can be provided to the SCK pin to feed the smart card clock inputs. This means that FLEX_US_MR.CLKO can be set.

This clock is then divided by the value programmed in the FI_DI_RATIO field in the FI DI Ratio Register (FLEX_US_FIDI). This is performed by the Sampling Divider, which performs a division by up to 65535 in ISO7816 mode. The noninteger values of the Fi/Di Ratio are not supported and the user must program the FI_DI_RATIO field to a value as close as possible to the expected value.

The FI_DI_RATIO field resets to the value 0x174 (372 in decimal) and is the most common divider between the ISO7816 clock and the bit rate (Fi = 372, Di = 1).

The following figure shows the relation between the Elementary Time Unit, corresponding to a bit time, and the ISO 7816 clock.

**Figure 46-4. Elementary Time Unit (ETU)**




### 46.7.2 Receiver and Transmitter Control

After reset, the receiver is disabled. The user must enable the receiver by setting the RXEN bit in the USART Control Register (FLEX_US_CR). However, the receiver registers can be programmed before the receiver clock is enabled.

After reset, the transmitter is disabled. The user must enable it by setting the TXEN bit in FLEX_US_CR. However, the transmitter registers can be programmed before being enabled.

The receiver and the transmitter can be enabled together or independently.

At any time, the software can perform a reset on the receiver or the transmitter of the USART by setting the corresponding bit, RSTRX and RSTTX respectively, in FLEX_US_CR. The software resets clear the status flag and reset internal state machines but the user interface configuration registers hold the value configured prior to software reset. Regardless of what the receiver or the transmitter is performing, the communication is immediately stopped.

The user can also independently disable the receiver or the transmitter by setting RXDIS and TXDIS respectively in FLEX_US_CR. If the receiver is disabled during a character reception, the USART waits until the end of reception of the current character, then the reception is stopped. If the transmitter is disabled while it is operating, the USART waits the end of transmission of both the current character and character being stored in the USART Transmit Holding Register (FLEX_US_THR). If a timeguard is programmed, it is handled normally.

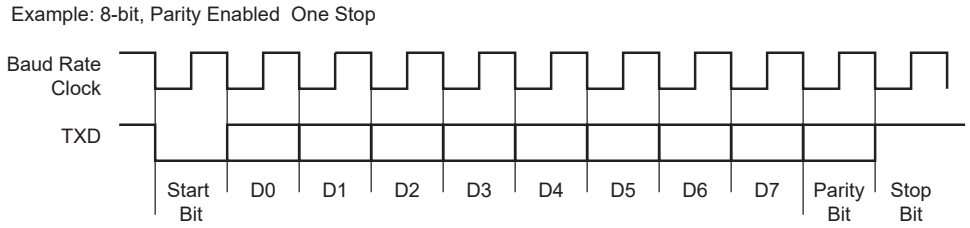
### 46.7.3 Synchronous and Asynchronous Modes

#### 46.7.3.1 Transmitter Operations

The transmitter performs the same in both Synchronous and Asynchronous operating modes (SYNC = 0 or SYNC = 1). One start bit, up to 9 data bits, 1 optional parity bit and up to 2 stop bits are successively shifted out on the TXD pin at each falling edge of the programmed serial clock.

The number of data bits is selected by the CHRL field and the MODE9 bit in FLEX_US_MR. Nine bits are selected by setting the MODE9 bit regardless of the CHRL field. The parity bit is set according to the PAR field in FLEX_US_MR. The even, odd, space, marked or none parity bit can be configured. The MSBF bit in FLEX_US_MR configures which data bit is sent first. If written to 1, the most significant bit is sent first. If written to 0, the less significant bit is sent first. The number of stop bits is selected by the NBSTOP field in FLEX_US_MR. The 1.5 stop bit is supported in Asynchronous mode only.

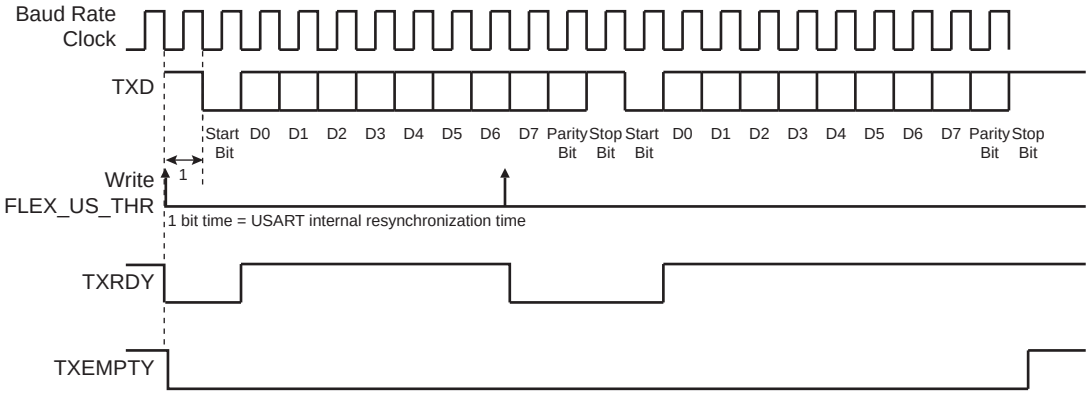
**Figure 46-5. Character Transmit**




The characters are sent by writing in FLEX_US_THR. The transmitter reports two status bits in the USART Channel Status Register (FLEX_US_CSR): TXRDY (Transmitter Ready), which indicates that FLEX_US_THR is empty and TXEMPTY, which indicates that all the characters written in FLEX_US_THR have been processed. When the current character processing is completed, the last character written in FLEX_US_THR is transferred into the shift register of the transmitter and FLEX_US_THR is emptied, thus TXRDY rises.

Both TXRDY and TXEMPTY bits are low when the transmitter is disabled. Writing a character in FLEX_US_THR while TXRDY is low has no effect and the written character is lost.

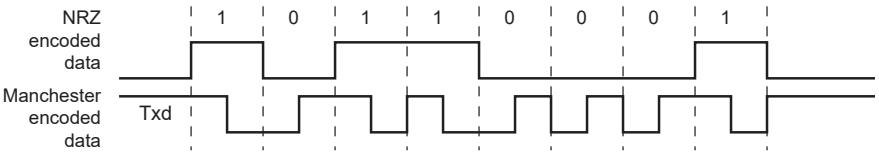
**Figure 46-6. Transmitter Status**




### 46.7.3.2 Manchester Encoder

When the Manchester encoder is in use, characters transmitted through the USART are encoded based on biphase Manchester II format. To enable this mode, set the FLEX_US_MR.MAN bit to 1. Depending on polarity configuration, a logic level (zero or one), is transmitted as a coded signal one-to-zero or zero-to-one. Thus, a transition always occurs at the midpoint of each bit time. It consumes more bandwidth than the original NRZ signal (2x) but the receiver has more error control since the expected input must show a change at the center of a bit cell. An example of Manchester encoded sequence is: the byte 0xB1 or 10110001 encodes to 10 01 10 10 01 01 01 10, assuming the default polarity of the encoder. The following figure illustrates this coding scheme.

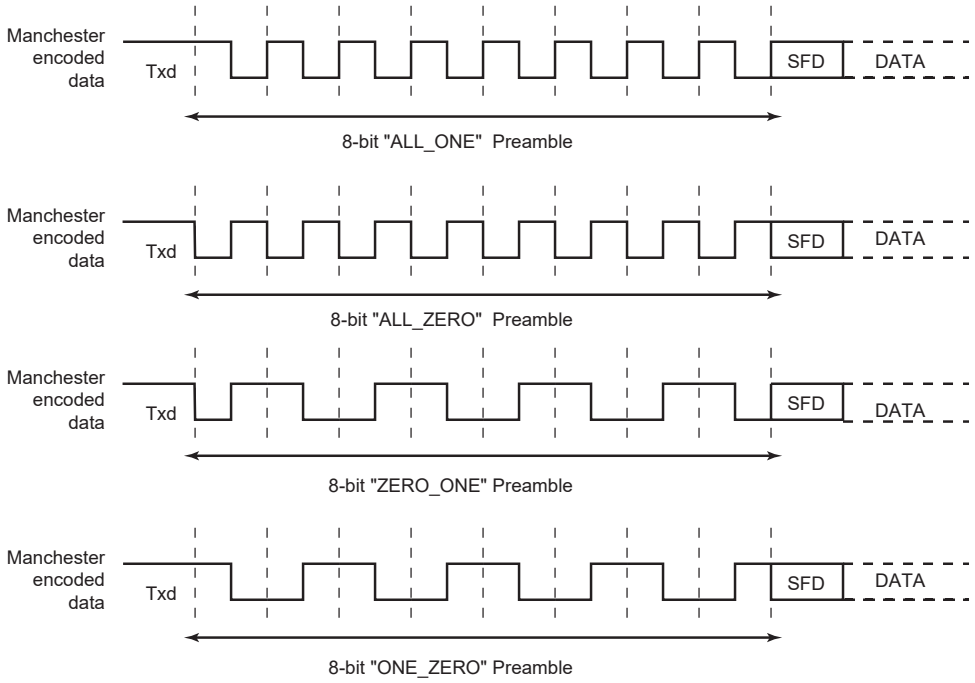
**Figure 46-7. NRZ to Manchester Encoding**




The Manchester encoded character can also be encapsulated by adding both a configurable preamble and a start frame delimiter pattern. Depending on the configuration, the preamble is a training sequence, composed of a predefined pattern with a programmable length from 1 to 15 bit times. If the preamble length is set to 0, the preamble waveform is not generated prior to any character. The preamble pattern is

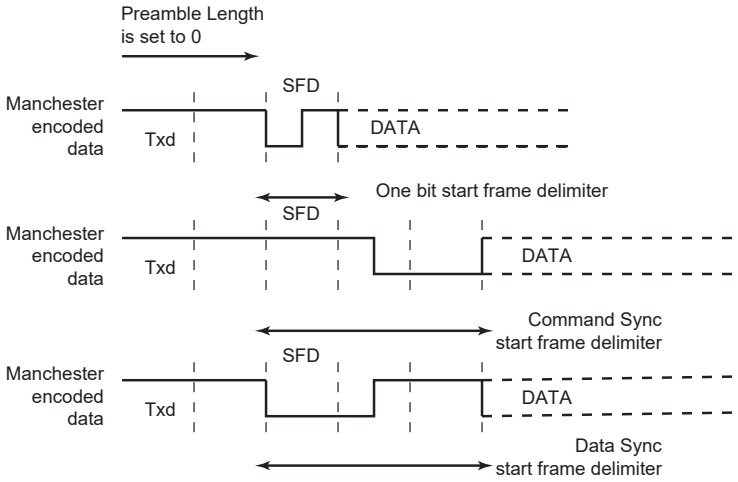
chosen among the following sequences: ALL_ONE, ALL_ZERO, ONE_ZERO or ZERO_ONE, writing the FLEX_US_MAN.TX_PP field. The TX_PL field is used to configure the preamble length. The following figure illustrates and defines the valid patterns. To improve flexibility, the encoding scheme can be configured using the FLEX_US_MAN.TX_MPOL bit. If the TX_MPOL bit is set to zero (default), a logic zero is encoded with a zero-to-one transition and a logic one is encoded with a one-to-zero transition. If the TX_MPOL bit is set to one, a logic one is encoded with a one-to-zero transition and a logic zero is encoded with a zero-to-one transition.

**Figure 46-8. Preamble Patterns, Default Polarity Assumed**




A start frame delimiter is to be configured using the FLEX_US_MR.ONEBIT bit. It consists of a user-defined pattern that indicates the beginning of a valid data. The following figure illustrates these patterns. If the start frame delimiter, also known as the start bit, is one bit, (ONEBIT = 1), a logic zero is Manchester encoded and indicates that a new character is being sent serially on the line. If the start frame delimiter is a synchronization pattern also referred to as sync (ONEBIT = 0), a sequence of three bit times is sent serially on the line to indicate the start of a new character. The sync waveform is in itself an invalid Manchester waveform as the transition occurs at the middle of the second bit time. Two distinct sync patterns are used: the command sync and the data sync. The command sync has a logic one level for one and a half bit times, then a transition to logic zero for the second one and a half bit times. If the FLEX_US_MR.MODSYNC bit is set to 1, the next character is a command. If it is set to 0, the next character is a data. When direct memory access is used, the MODSYNC bit can be immediately updated with a modified character located in memory. To enable this mode, the FLEX_US_MR.VAR_SYNC bit must be set. In this case, the FLEX_US_MR.MODSYNC bit is bypassed and the sync configuration is held in the FLEX_US_THR.TXSYNH bit. The USART character format is modified and includes sync information.

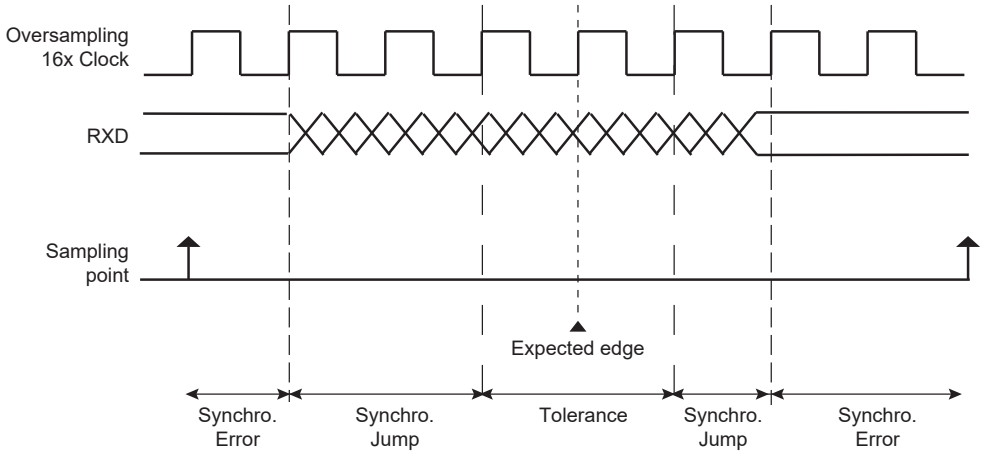
**Figure 46-9. Start Frame Delimiter**




#### 46.7.3.2.1 Drift Compensation

Drift compensation is available only in 16X Oversampling mode. An hardware recovery system allows a larger clock drift. To enable the hardware system, the bit in the FLEX_US_MAN register must be set. If the RXD edge is one 16X clock cycle from the expected edge, this is considered as normal jitter and no corrective actions is taken. If the RXD event is between 4 and 2 clock cycles before the expected edge, then the current period is shortened by one clock cycle. If the RXD event is between 2 and 3 clock cycles after the expected edge, then the current period is lengthened by one clock cycle. These intervals are considered to be drift and so corrective actions are automatically taken.

**Figure 46-10. Bit Resynchronization**




### 46.7.3.3 Asynchronous Receiver

If the USART is programmed in Asynchronous operating mode (SYNC = 0), the receiver oversamples the RXD input line. The oversampling is either 16 or 8 times the baud rate clock, depending on the FLEX_US_MR.OVER bit.

The receiver samples the RXD line. If the line is sampled during one half of a bit time to 0, a start bit is detected and data, parity and stop bits are successively sampled on the bit rate clock.

If the oversampling is 16 (OVER = 0), a start is detected at the eighth sample to 0. Data bits, parity bit and stop bit are assumed to have a duration corresponding to 16 oversampling clock cycles. If the oversampling is 8 (OVER = 1), a start bit is detected at the fourth sample to 0. Data bits, parity bit and stop bit are assumed to have a duration corresponding to 8 oversampling clock cycles.

The number of data bits, first bit sent and Parity mode are selected by the same fields and bits as the transmitter, i.e., respectively CHRL, MODE9, MSBF and PAR. For the synchronization mechanism only, the number of stop bits has no effect on the receiver as it considers only one stop bit, regardless of the NBSTOP field, so that resynchronization between the receiver and the transmitter can occur. Moreover, as soon as the stop bit is sampled, the receiver starts looking for a new start bit so that resynchronization can also be accomplished when the transmitter is operating with one stop bit.

The following figures illustrate start detection and character reception when USART operates in Asynchronous mode.

**Figure 46-11. Asynchronous Start Detection**

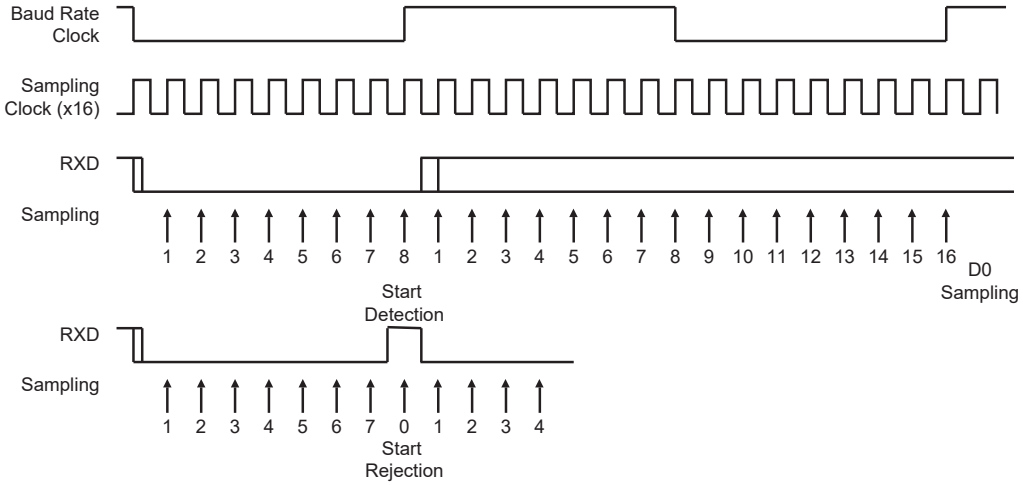
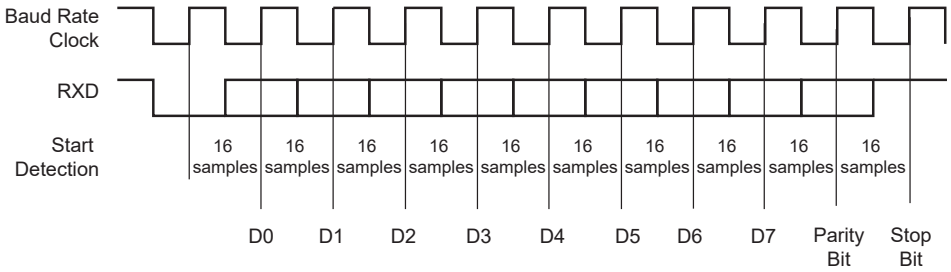



**Figure 46-12. Asynchronous Character Reception**

Example: 8-bit, Parity Enabled




### 46.7.3.4 Manchester Decoder

When the FLEX_US_MR.MAN bit is set, the Manchester decoder is enabled. The decoder performs both preamble and start frame delimiter detection. One input line is dedicated to Manchester encoded input data.

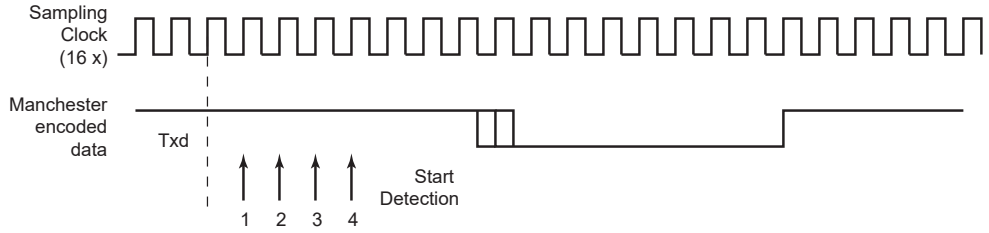
An optional preamble sequence can be defined. Its length is user-defined and totally independent of the transmitter side. Use the FLEX_US_MAN.RX_PL field to configure the length of the preamble sequence. If the length is set to 0, no preamble is detected and the function is disabled. In addition, the polarity of the input stream is programmable with the FLEX_US_MAN.RX_MPOL bit. Depending on the desired application, the preamble pattern matching is to be defined via the FLEX_US_MAN.RX_PP field. See figure Preamble Patterns, Default Polarity Assumed for available preamble patterns.

Unlike preamble, the start frame delimiter is shared between Manchester Encoder and Decoder. So, if ONEBIT bit = 1, only a zero encoded Manchester can be detected as a valid start frame delimiter. If ONEBIT = 0, only a sync pattern is detected as a valid start frame delimiter. Decoder operates by

detecting transition on incoming stream. If RXD is sampled during one quarter of a bit time to zero, a start bit is detected. See the following figure. The sample pulse rejection mechanism applies.

The FLEX_US_MAN.RXIDLEV bit informs the USART of the receiver line idle state value (receiver line inactive). The user must define RXIDLEV to ensure reliable synchronization. By default, RXIDLEV is set to one (receiver line is at level 1 when there is no activity).

**Figure 46-13. Asynchronous Start Bit Detection**




The receiver is activated and starts preamble and frame delimiter detection, sampling the data at one quarter and then three quarters. If a valid preamble pattern or start frame delimiter is detected, the receiver continues decoding with the same synchronization. If the stream does not match a valid pattern or a valid start frame delimiter, the receiver resynchronizes on the next valid edge.The minimum time threshold to estimate the bit value is three quarters of a bit time.

If a valid preamble (if used) followed with a valid start frame delimiter is detected, the incoming stream is decoded into NRZ data and passed to USART for processing. The following figure illustrates Manchester pattern mismatch. When incoming data stream is passed to the USART, the receiver is also able to detect Manchester code violation. A code violation is a lack of transition in the middle of a bit cell. In this case, the MANE flag in FLEX_US_CSR is raised. It is cleared by writing a one to FLEX_US_CR.RSTSTA. See figure "Manchester Error Flag" below for an example of Manchester error detection during the data phase.

**Figure 46-14. Preamble Pattern Mismatch**

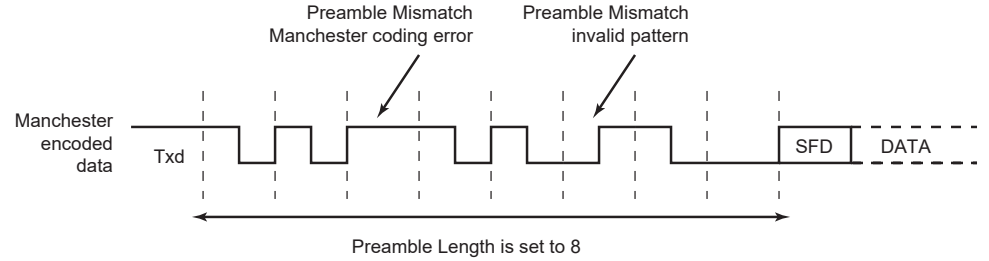
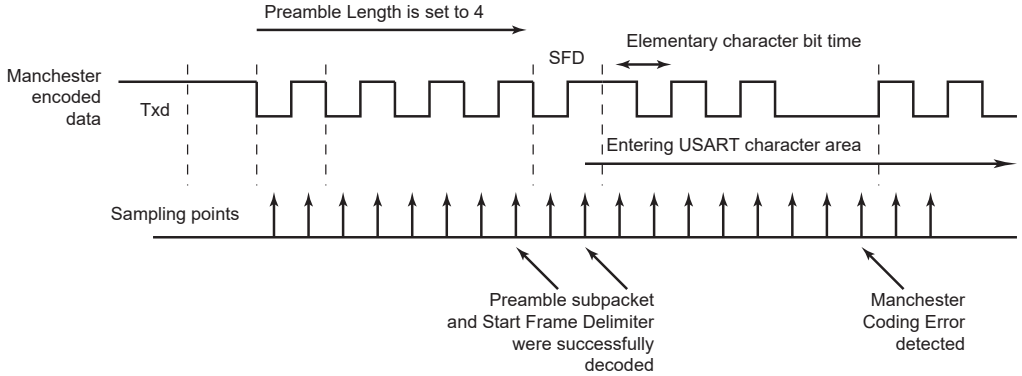



**Figure 46-15. Manchester Error Flag**



When the start frame delimiter is a sync pattern (ONEBIT = 0), both command and data delimiter are supported. If a valid sync is detected, the received character is written as RXCHR field in the Receive Holding Register (FLEX_US_RHR) and the RXSYNH is updated. RXCHR is set to 1 when the received character is a command, and it is set to 0 if the received character is a data. This mechanism alleviates and simplifies the direct memory access as the character contains its own sync field in the same register.

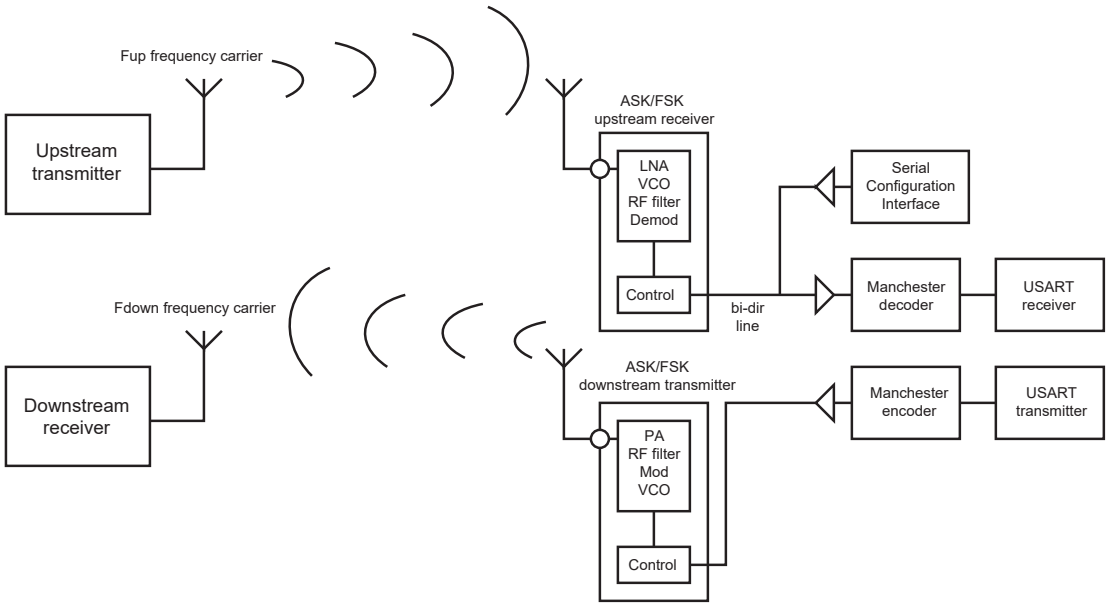
As the decoder is setup to be used in Unipolar mode, the first bit of the frame has to be a zero-to-one transition.

#### 46.7.3.5 Radio Interface: Manchester Encoded USART Application

This section describes low data rate RF transmission systems and their integration with a Manchester encoded USART. These systems are based on transmitter and receiver ICs that support ASK and FSK modulation schemes.

The goal is to perform full-duplex radio transmission of characters using two different frequency carriers. See configuration in the following figure.

**Figure 46-16. Manchester Encoded Characters RF Transmission**




The USART peripheral is configured as a Manchester encoder/decoder. Looking at the downstream communication channel, Manchester encoded characters are serially sent to the RF transmitter. This may also include a user defined preamble and a start frame delimiter. Mostly, preamble is used in the RF receiver to distinguish between a valid data from a transmitter and signals due to noise. The Manchester stream is then modulated. See the following figure for an example of ASK modulation scheme. When a logic one is sent to the ASK modulator, the power amplifier, referred to as PA, is enabled and transmits an RF signal at downstream frequency. When a logic zero is transmitted, the RF signal is turned off. If the FSK modulator is activated, two different frequencies are used to transmit data. When a logic 1 is sent, the modulator outputs an RF signal at frequency F0 and switches to F1 if the data sent is a 0. See figure "FSK Modulator Output" below.

From the receiver side, another carrier frequency is used. The RF receiver performs a bit check operation examining demodulated data stream. If a valid pattern is detected, the receiver switches to Receiving mode. The demodulated stream is sent to the Manchester decoder. Because of bit checking inside RF IC, the data transferred to the microcontroller is reduced by a user-defined number of bits. The Manchester preamble length is to be defined in accordance with the RF IC configuration.

**Figure 46-17. ASK Modulator Output**

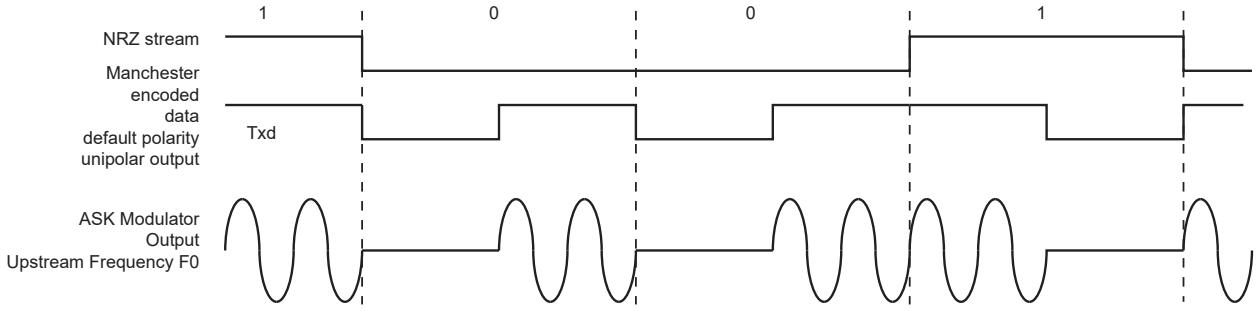
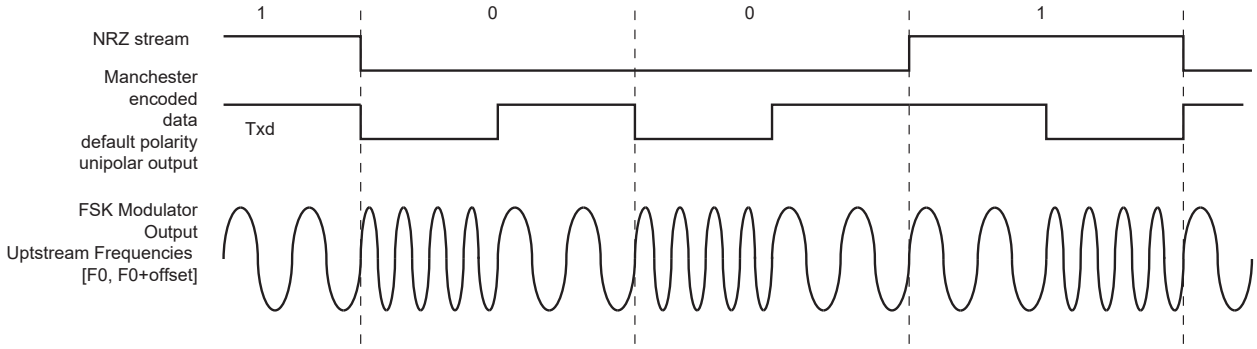



**Figure 46-18. FSK Modulator Output**




### 46.7.3.6 Synchronous Receiver

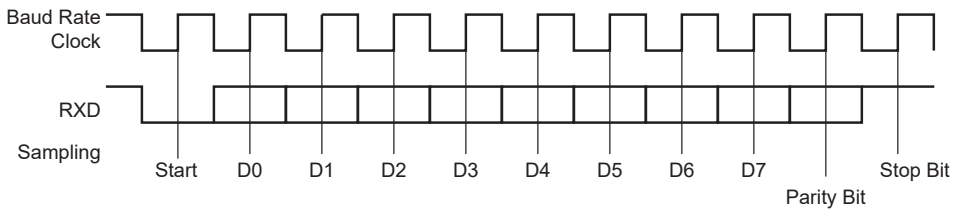
In Synchronous mode (SYNC = 1), the receiver samples the RXD signal on each rising edge of the baud rate clock. If a low level is detected, it is considered as a start. All data bits, the parity bit and the stop bits are sampled and the receiver waits for the next start bit. Synchronous mode operations provide a high-speed transfer capability.

Configuration fields and bits are the same as in Asynchronous mode.

The following figure illustrates a character reception in Synchronous mode.

**Figure 46-19. Synchronous Mode Character Reception**

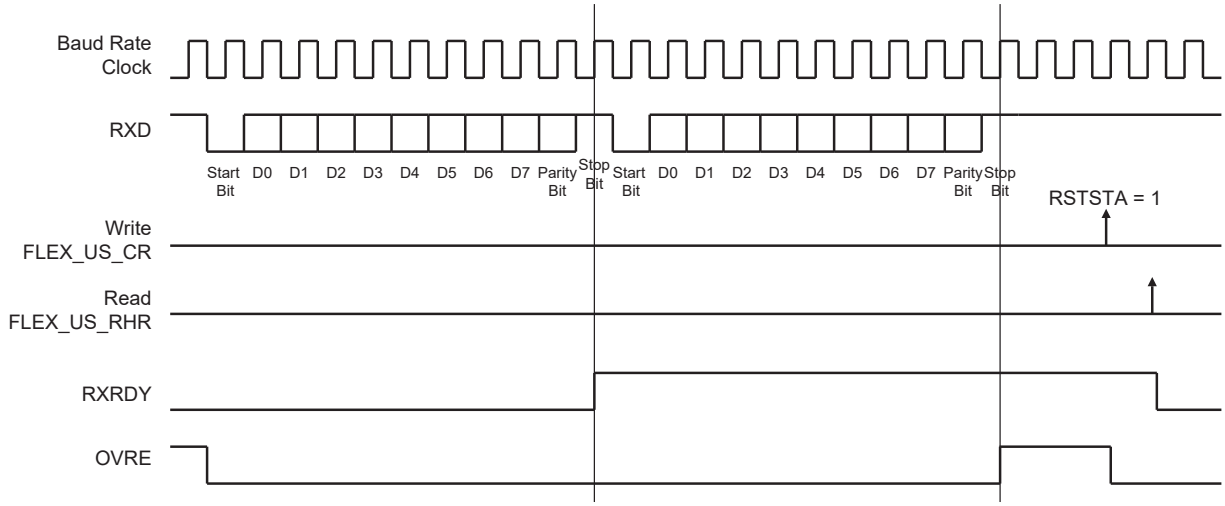
Example: 8-bit, Parity Enabled 1 Stop




### 46.7.3.7 Receiver Operations

When a character reception is completed, it is transferred to the Receive Holding Register (FLEX_US_RHR) and the FLEX_US_CSR.RXRDY bit is raised. If a character is completed while the RXRDY is set, the Overrun Error (OVRE) bit is set. The last character is transferred into FLEX_US_RHR and overwrites the previous one. The OVRE bit is cleared by writing a one to Reset Status bit FLEX_US_CR.RSTSTA.

**Figure 46-20. Receiver Status**




### 46.7.3.8 Parity

The USART supports five parity modes that are selected by writing to the FLEX_US_MR.PAR field. The PAR field also enables the Multidrop mode (see section Multidrop Mode). Even and odd parity bit generation and error detection are supported.

If even parity is selected, the parity generator of the transmitter drives the parity bit to 0 if a number of 1s in the character data bit is even, and to 1 if the number of 1s is odd. Accordingly, the receiver parity checker counts the number of received 1s and reports a parity error if the sampled parity bit does not correspond. If odd parity is selected, the parity generator of the transmitter drives the parity bit to 1 if a number of 1s in the character data bit is even, and to 0 if the number of 1s is odd. Accordingly, the receiver parity checker counts the number of received 1s and reports a parity error if the sampled parity bit does not correspond. If the mark parity is used, the parity generator of the transmitter drives the parity bit to 1 for all characters. The receiver parity checker reports an error if the parity bit is sampled to 0. If the space parity is used, the parity generator of the transmitter drives the parity bit to 0 for all characters. The receiver parity checker reports an error if the parity bit is sampled to 1. If parity is disabled, the transmitter does not generate any parity bit and the receiver does not report any parity error.

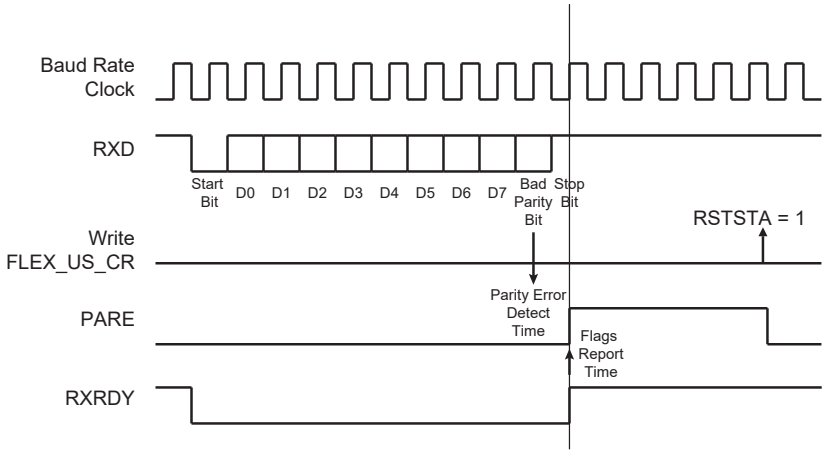
The following table shows an example of the parity bit for the character 0x41 (character ASCII “A”) depending on the configuration of the USART. Because there are two bits set to 1 in the character value, the parity bit is set to 1 when the parity is odd, or configured to 0 when the parity is even.

**Table 46-7. Parity Bit Examples**

| Character | Hexadecimal | Binary | Parity Bit | Parity Mode |
|---|---|---|---|---|
| A | 0x41 | 0100 0001 | 1 | Odd |
| A | 0x41 | 0100 0001 | 0 | Even |
| A | 0x41 | 0100 0001 | 1 | Mark |
| A | 0x41 | 0100 0001 | 0 | Space |
| A | 0x41 | 0100 0001 | None | None |

When the receiver detects a parity error, it sets the Parity Error bit FLEX_US_CSR.PARE. The PARE bit can be cleared by writing a one to the FLEX_US_CR.RSTSTA bit. The following figure illustrates the parity bit status setting and clearing.

**Figure 46-21. Parity Error**




#### 46.7.3.9 Multidrop Mode

If the value 0x6 or 0x07 is written to the FLEX_US_MR.PAR field, the USART runs in Multidrop mode. This mode differentiates the data characters and the address characters. Data are transmitted with the parity bit to 0 and addresses are transmitted with the parity bit to 1.

If the USART is configured in Multidrop mode, the receiver sets the PARE parity error bit when the parity bit is high and the transmitter is able to send a character with the parity bit high when a one is written to the FLEX_US_CR.SENDA bit.

To handle parity error, the PARE bit is cleared by writing a one to the FLEX_US_CR.RSTSTA bit.

The transmitter sends an address byte (parity bit set) when the FLEX_US_CR.SENDA bit is written to 1. In this case, the next byte written to FLEX_US_THR is transmitted as an address. Any character written in FLEX_US_THR when the SENDA command is not written is transmitted normally with parity to 0.

#### 46.7.3.10 Transmitter Timeguard

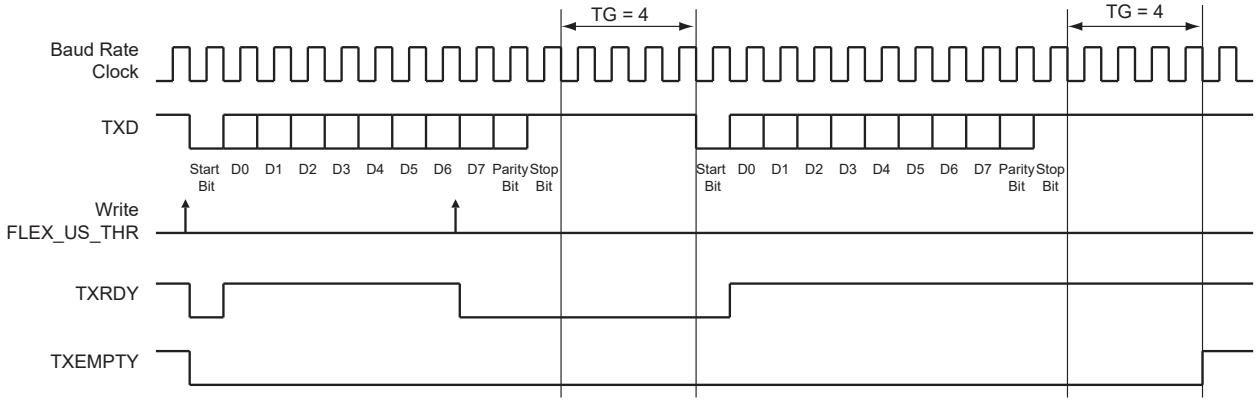
The timeguard feature enables the USART interface with slow remote devices.

The timeguard function enables the transmitter to insert an idle state on the TXD line between two characters. This idle state actually acts as a long stop bit.

The duration of the idle state is programmed in the TG field of the Transmitter Timeguard Register (FLEX_US_TTGR). When this field is written to zero, no timeguard is generated. Otherwise, the transmitter holds a high level on TXD after each transmitted byte during the number of bit periods programmed in TG in addition to the number of stop bits.

As illustrated in the following figure, the behavior of the TXRDY and TXEMPTY status bits is modified by the programming of a timeguard. TXRDY rises only when the start bit of the next character is sent, and thus remains to 0 during the timeguard transmission if a character has been written in FLEX_US_THR. TXEMPTY remains low until the timeguard transmission is completed as the timeguard is part of the current character being transmitted.

**Figure 46-22. Timeguard Operations**




The following table indicates the maximum length of a timeguard period that the transmitter can handle in relation to the function of the baud rate.

**Table 46-8. Maximum Timeguard Length Depending on Baud Rate**

| Baud Rate (bit/s) | Bit Time (µs) | Timeguard (ms) |
|---|---|---|
| 1,200 | 833 | 212.50 |
| 9,600 | 104 | 26.56 |
| 14,400 | 69.4 | 17.71 |
| 19,200 | 52.1 | 13.28 |
| 28,800 | 34.7 | 8.85 |
| 38,400 | 26 | 6.63 |
| 56,000 | 17.9 | 4.55 |
| 57,600 | 17.4 | 4.43 |
| 115,200 | 8.7 | 2.21 |

#### 46.7.3.11 Receiver Timeout

The Receiver Timeout provides support in handling variable-length frames. This feature detects an idle condition on the RXD line. When a timeout is detected, the FLEX_US_CSR.TIMEOUT bit rises and can generate an interrupt, thus indicating to the driver an end of frame.

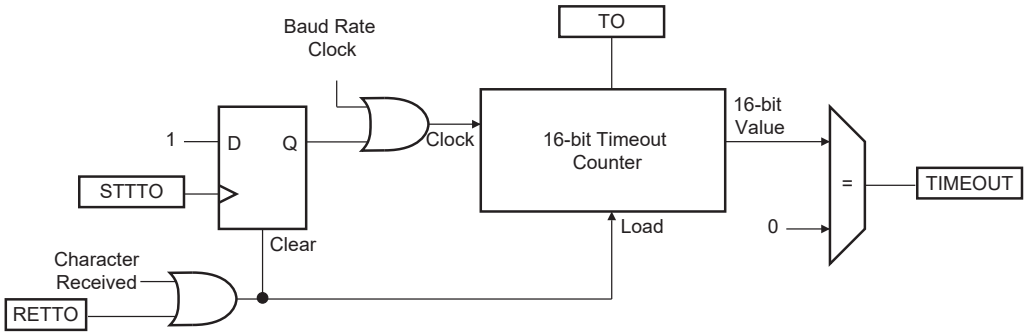
The timeout delay period (during which the receiver waits for a new character) is programmed in the TO field of the Receiver Timeout Register (FLEX_US_RTOR). If the TO field is written to 0, the Receiver Timeout is disabled and no timeout is detected. The FLEX_US_CSR.TIMEOUT bit remains at 0. Otherwise, the receiver loads a 16-bit counter with the value programmed in TO. This counter is decremented at each bit period and reloaded each time a new character is received. If the counter reaches 0, the FLEX_US_CSR.TIMEOUT bit rises. Then, the user can either:

- Stop the counter clock until a new character is received. This is performed by writing a ‘1’ to FLEX_US_CR.STTTO. In this case, the idle state on RXD before a new character is received does not provide a timeout. This prevents having to handle an interrupt before a character is received and enables waiting for the next idle state on RXD after a frame is received.
- Obtain an interrupt while no character is received. This is performed by writing a ‘1’ to FLEX_US_CR.RETTO. In this case, the counter starts counting down immediately from the value

TO. This generates a periodic interrupt so that a user timeout can be handled, for example when no key is pressed on a keyboard.

The following figure shows the block diagram of the Receiver Timeout feature.

**Figure 46-23. Receiver Timeout Block Diagram**




The following table gives the maximum timeout period for some standard baud rates.

**Table 46-9. Maximum Timeout Period**

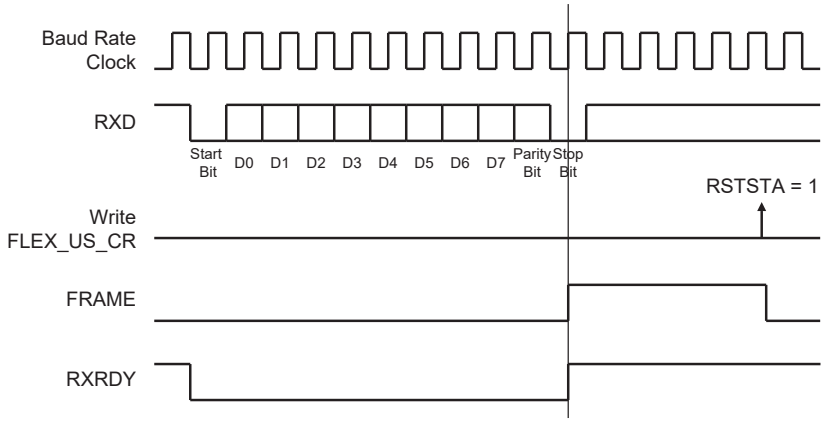
| Baud Rate (bit/s) | Bit Time (µs) | Timeout (ms) |
|---|---|---|
| 600 | 1,667 | 109,225 |
| 1,200 | 833 | 54,613 |
| 2,400 | 417 | 27,306 |
| 4,800 | 208 | 13,653 |
| 9,600 | 104 | 6,827 |
| 14,400 | 69 | 4,551 |
| 19,200 | 52 | 3,413 |
| 28,800 | 35 | 2,276 |
| 38,400 | 26 | 1,704 |
| 56,000 | 18 | 1,170 |
| 57,600 | 17 | 1,138 |
| 200,000 | 5 | 328 |

#### 46.7.3.12 Framing Error

The receiver is capable of detecting framing errors. A framing error happens when the stop bit of a received character is detected at level 0. This can occur if the receiver and the transmitter are fully desynchronized.

A framing error is reported on the FLEX_US_CSR.FRAME bit. The FRAME bit is asserted in the middle of the stop bit as soon as the framing error is detected. It is cleared by writing a one to the FLEX_US_CR.RSTSTA bit.

**Figure 46-24. Framing Error Status**




#### 46.7.3.13 Transmit Break

The user can request the transmitter to generate a break condition on the TXD line. A break condition drives the TXD line low during at least one complete character. It appears the same as a 0x00 character sent with the parity and the stop bits to 0. However, the transmitter holds the TXD line at least during one character until the user requests the break condition to be removed.

A break is transmitted by setting the FLEX_US_CR.STTBRK bit. This can be done at any time, either while the transmitter is empty (no character in either the shift register or in FLEX_US_THR) or when a character is being transmitted. If a break is requested while a character is being shifted out, the character is first completed before the TXD line is held low.

Once the Start Break command is requested, further Start Break commands are ignored until the end of the break is completed.

The break condition is removed by setting the FLEX_US_CR.STPBRK bit. If the Stop Break command is requested before the end of the minimum break duration (one character, including start, data, parity and stop bits), the transmitter ensures that the break condition completes.

The transmitter considers the break as though it is a character, i.e., the Start Break and Stop Break commands are processed only if the FLEX_US_CSR.TXRDY bit = 1 and the start of the break condition clears the TXRDY and TXEMPTY bits as if a character was processed.

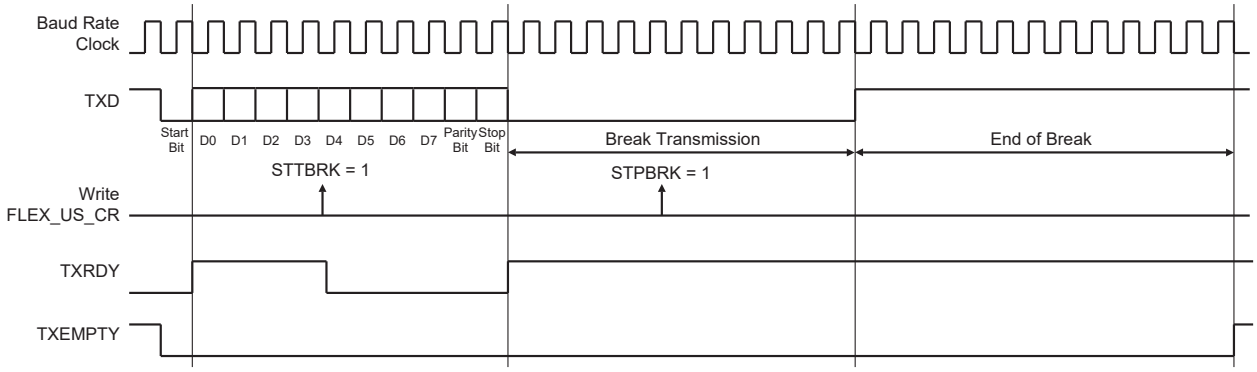
Setting both the FLEX_US_CR.STTBRK and FLEX_US_CR.STPBRK bits can lead to an unpredictable result. All Stop Break commands requested without a previous Start Break command are ignored. A byte written into the Transmit Holding register while a break is pending, but not started, is ignored.

After the break condition, the transmitter returns the TXD line to 1 for a minimum of 12 bit times. Thus, the transmitter ensures that the remote receiver detects correctly the end of break and the start of the next character. If the timeguard is programmed with a value higher than 12, the TXD line is held high for the timeguard period.

After holding the TXD line for this period, the transmitter resumes normal operations.

The following figure illustrates the effect of both the Start Break (STTBRK) and Stop Break (STPBRK) commands on the TXD line.

**Figure 46-25. Break Transmission**




#### 46.7.3.14 Receive Break

The receiver detects a break condition when all data, parity and stop bits are low. This corresponds to detecting a framing error with data to 0x00, but FRAME remains low.

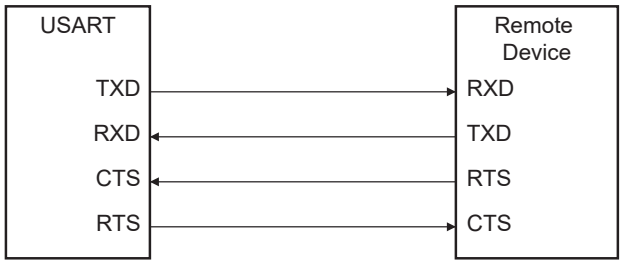
When the low stop bit is detected, the receiver asserts the FLEX_US_CSR.RXBRK bit. FLEX_US_CSR.RXBRK may be cleared by setting the FLEX_US_CR.RSTSTA bit.

An end of receive break is detected by a high level for at least 2/16ths of a bit period in Asynchronous operating mode or one sample at high level in Synchronous operating mode. The end of break detection also asserts the RXBRK bit.

#### 46.7.3.15 Hardware Handshaking

The USART features a hardware handshaking out-of-band flow control. The RTS and CTS pins are used to connect with the remote device, as shown in the following figure.

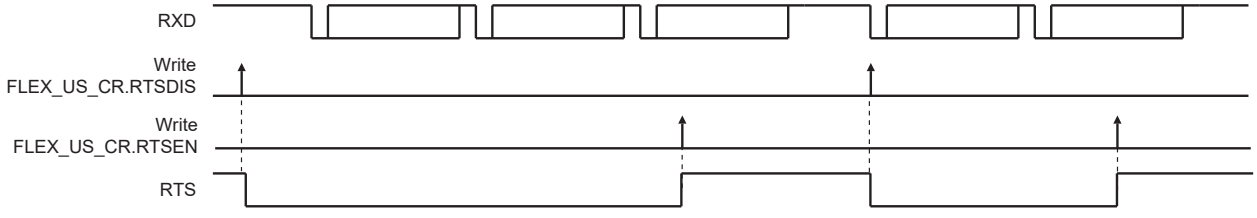
**Figure 46-26. Connection with a Remote Device for Hardware Handshaking**




Setting the USART to operate with hardware handshaking is performed by writing the FLEX_US_MR.USART_MODE field to the value 0x2.

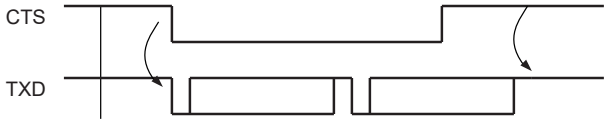
The USART behavior when hardware handshaking is enabled is the same as the behavior in standard Synchronous or Asynchronous mode, except that the receiver drives the RTS pin as described below and the level on the CTS pin modifies the behavior of the transmitter as described below. Using this mode requires using the DMAC channel for reception. The transmitter can handle hardware handshaking in any case.

**Figure 46-27. RTS Line Software Control when FLEX_US_MR.USART_MODE = 2**



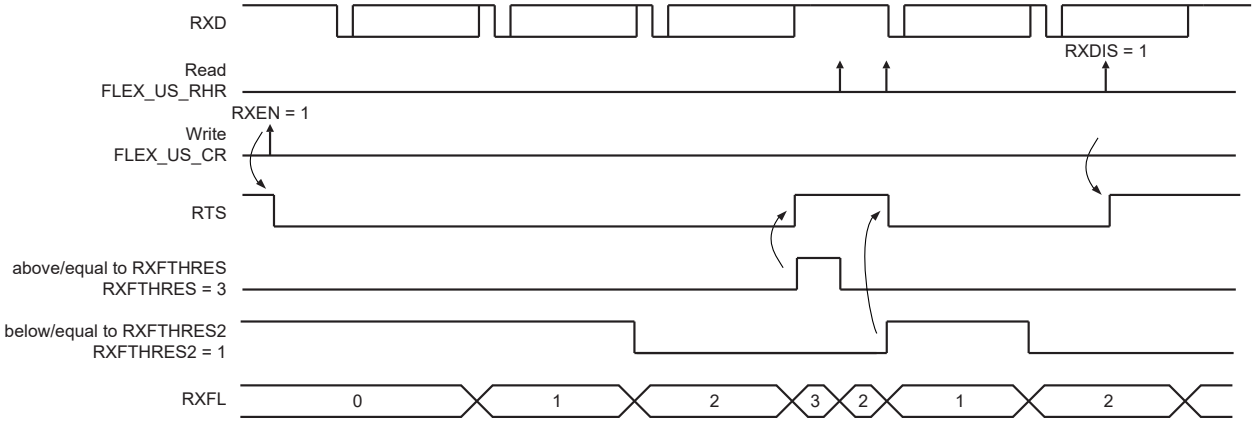
The following figure shows how the transmitter operates if hardware handshaking is enabled. The CTS pin disables the transmitter. If a character is being processed, the transmitter is disabled only after the completion of the current character and transmission of the next character happens as soon as the pin CTS falls.

**Figure 46-28. Transmitter Behavior when Operating with Hardware Handshaking**




If USART FIFOs are enabled (bit FLEX_US_CR.FIFOEN), the RTS pin can be controlled by the USART Receive FIFO thresholds. The RTS pin control through Receive FIFO thresholds can be activated with the FLEX_US_FMR.FRTSC bit. Once activated, the RTS pin will be controlled by Receive FIFO thresholds, set to level 1 each time RXFTHRES is reached and set to level '0' each time RXFTHRES2 is reached (and RXFTHRES is not reached).

**Figure 46-29. Receiver Behavior When FIFO Enabled and FRTSC Set to '1'**




**Note:** In this mode, RXFTHRES must be > RXFTHRES2.

### 46.7.4 ISO7816 Mode

The USART features an ISO7816-compatible operating mode. This mode permits interfacing with smart cards and Security Access Modules (SAM) communicating through an ISO7816 link. Both T = 0 and T = 1 protocols defined by the ISO7816 specification are supported.

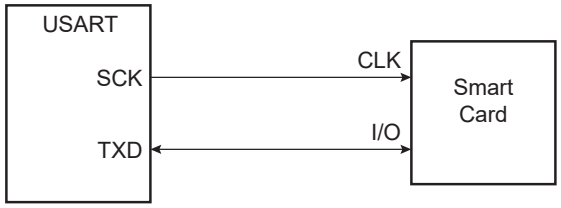
Setting the USART in ISO7816 mode is performed by writing the FLEX_US_MR.USART_MODE field to the value 0x4 for protocol T = 0 and to the value 0x6 for protocol T = 1.

#### 46.7.4.1 ISO7816 Mode Overview

The ISO7816 is a half-duplex communication on only one bidirectional line. The baud rate is determined by a division of the clock provided to the remote device (see figure in section Baud Rate Generator).

The USART connects to a smart card as shown in the following figure. The TXD line becomes bidirectional and the baud rate generator feeds the ISO7816 clock on the SCK pin. As the TXD pin becomes bidirectional, its output remains driven by the output of the transmitter but only when the transmitter is active while its input is directed to the input of the receiver. The USART is considered as the master of the communication as it generates the clock.

**Figure 46-30. Connection of a Smart Card to the USART**




When operating in ISO7816, either in T = 0 or T = 1 modes, the character format is partially predefined. The configuration is forced to 8 data bits, and 1 or 2 stop bits, regardless of the values programmed in the CHRL, MODE9 and CHMODE fields. MSBF can be used to transmit LSB or MSB first. The bit INVDATA can be used to transmit in Normal or Inverse mode.

The USART cannot operate concurrently in both Receiver and Transmitter modes as the communication is unidirectional at a time. It has to be configured according to the required mode by enabling or disabling either the receiver or the transmitter as desired. Enabling both the receiver and the transmitter at the same time in ISO7816 mode may lead to unpredictable results.

The ISO7816 specification defines an inverse transmission format. Data bits of the character must be transmitted on the I/O line at their negative value.

#### 46.7.4.2 Protocol T = 0

In T = 0 protocol, a character is made up of 1 start bit, 8 data bits, 1 parity bit and 1 guard time, which lasts two bit times. The transmitter shifts out the bits and does not drive the I/O line during the guard time.

If no parity error is detected, the I/O line remains at 1 during the guard time and the transmitter can continue with the transmission of the next character, as shown in the following figure.

If a parity error is detected by the receiver, it drives the I/O line to 0 during the guard time, as shown in figure "T = 0 Protocol with Parity Error" below. This error bit is also named NACK, for Non Acknowledge. In this case, the character lasts 1 bit time more, as the guard time length is the same and is added to the error bit time which lasts 1 bit time.

When the USART is the receiver and it detects an error, it does not load the erroneous character in the Receive Holding Register (FLEX_US_RHR). It appropriately sets the PARE bit in the Status Register (FLEX_US_CSR) so that the software can handle the error.

**Figure 46-31. T = 0 Protocol without Parity Error**

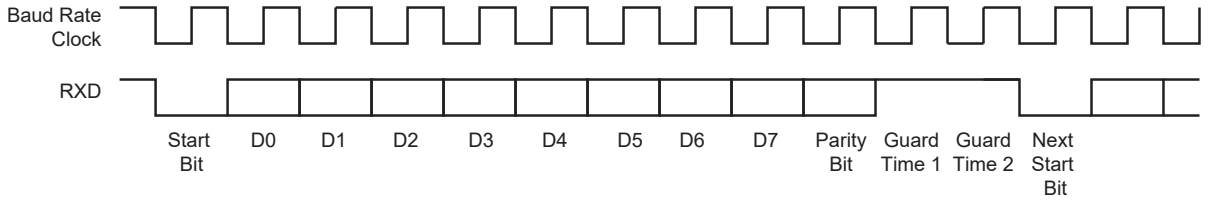
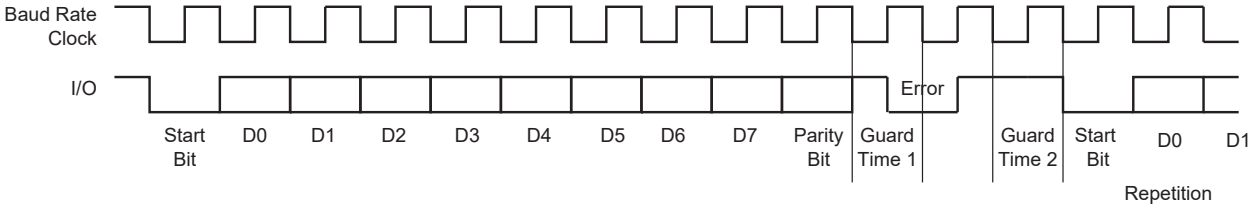



**Figure 46-32. T = 0 Protocol with Parity Error**



#### 46.7.4.2.1 Receive Error Counter

The USART receiver also records the total number of errors. This can be read in the Number of Error (FLEX_US_NER) register. The NB_ERRORS field can record up to 255 errors. Reading FLEX_US_NER automatically clears the NB_ERRORS field.

#### 46.7.4.2.2 Receive NACK Inhibit

The USART can be configured to inhibit an error. This is done by writing a ‘1’ to FLEX_US_MR.INACK. In this case, no error signal is driven on the I/O line even if a parity bit is detected.

Moreover, if INACK = 1, the erroneous received character is stored in the Receive Holding register as if no error occurred, and the RXRDY bit rises.

#### 46.7.4.2.3 Transmit Character Repetition

When the USART is transmitting a character and gets a NACK, it can automatically repeat the character before moving on to the next one. Repetition is enabled by writing the FLEX_US_MR.MAX_ITERATION field at a value higher than 0. Each character can be transmitted up to eight times: the first transmission plus seven repetitions.

If MAX_ITERATION does not equal zero, the USART repeats the character as many times as the value loaded in MAX_ITERATION.

When the USART repetition number reaches MAX_ITERATION, and the last repeated character is not acknowledged, the FLEX_US_CSR.ITER bit is set. If the repetition of the character is acknowledged by the receiver, the repetitions are stopped and the iteration counter is cleared.

The FLEX_US_CSR.ITER bit can be cleared by writing the FLEX_US_CR.RSTIT bit to 1.

#### 46.7.4.2.4 Disable Successive Receive NACK

The receiver can limit the number of successive NACKs sent back to the remote transmitter. This is programmed by setting the FLEX_US_MR.DSNACK bit. The maximum number of NACKs transmitted is programmed in the MAX_ITERATION field. As soon as MAX_ITERATION is reached, no error signal is driven on the I/O line and the FLEX_US_CSR.ITER bit is set.

### 46.7.4.3 Protocol T = 1

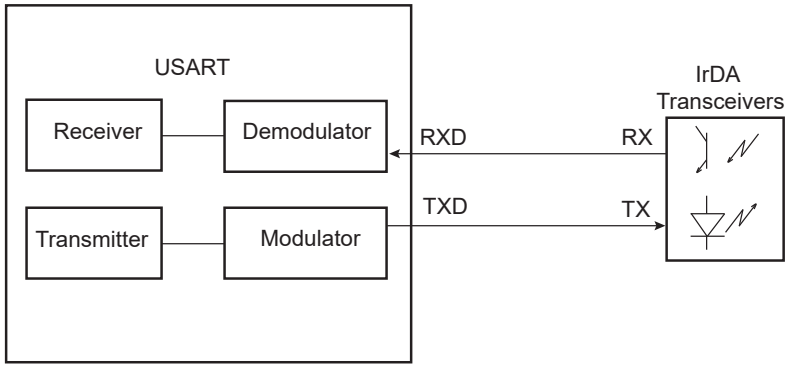
When operating in ISO7816 protocol T = 1, the transmission is similar to an asynchronous format with only one stop bit. The parity is generated when transmitting and checked when receiving. Parity error detection sets the FLEX_US_CSR.PARE bit.

## 46.7.5 IrDA Mode

The USART features an IrDA mode supplying half-duplex point-to-point wireless communication. It embeds the modulator and demodulator which allows a glueless connection to the infrared transceivers, as shown in the following figure. The modulator and demodulator are compliant with the IrDA specification version 1.1 and support data transfer speeds ranging from 2.4 kbit/s to 115.2 kbit/s.

The USART IrDA mode is enabled by setting the FLEX_US_MR.USART_MODE field to the value 0x8. The IrDA Filter Register (FLEX_US_IF) allows configuring the demodulator filter. The USART transmitter and receiver operate in a normal Asynchronous mode and all parameters are accessible. Note that the modulator and the demodulator are activated.

**Figure 46-33. Connection to IrDA Transceivers**




The receiver and the transmitter must be enabled or disabled according to the direction of the transmission to be managed.

To receive IrDA signals, the following needs to be done:

- Disable TX and Enable RX
- Configure the TXD pin as PIO and set it as an output to 0 (to avoid LED transmission). Disable the internal pullup (better for power consumption).
- Receive data

#### 46.7.5.1 IrDA Modulation

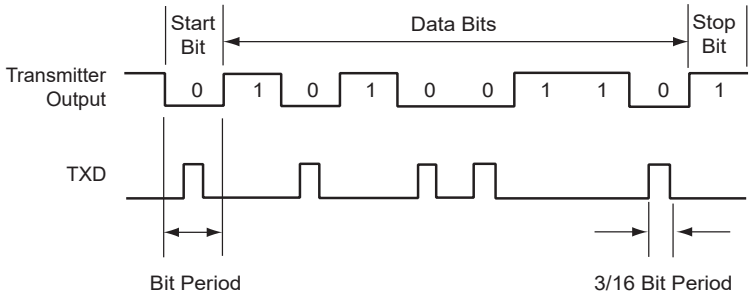
For baud rates up to and including 115.2 kbit/s, the RZI modulation scheme is used. “0” is represented by a light pulse of 3/16th of a bit time. Some examples of signal pulse duration are shown in the following table.

**Table 46-10. IrDA Pulse Duration**

| Baud Rate | Pulse Duration (3/16) |
|---|---|
| 2.4 kbit/s | 78.13 µs |
| 9.6 kbit/s | 19.53 µs |
| 19.2 kbit/s | 9.77 µs |
| 38.4 kbit/s | 4.88 µs |
| 57.6 kbit/s | 3.26 µs |
| 115.2 kbit/s | 1.63 µs |

The following figure shows an example of character transmission.

**Figure 46-34. IrDA Modulation**



#### 46.7.5.2 IrDA Baud Rate

The following table gives some examples of CD values, baud rate error and pulse duration. Note that the requirement on the maximum acceptable error of ±1.87% must be met.

**Table 46-11. IrDA Baud Rate Error**

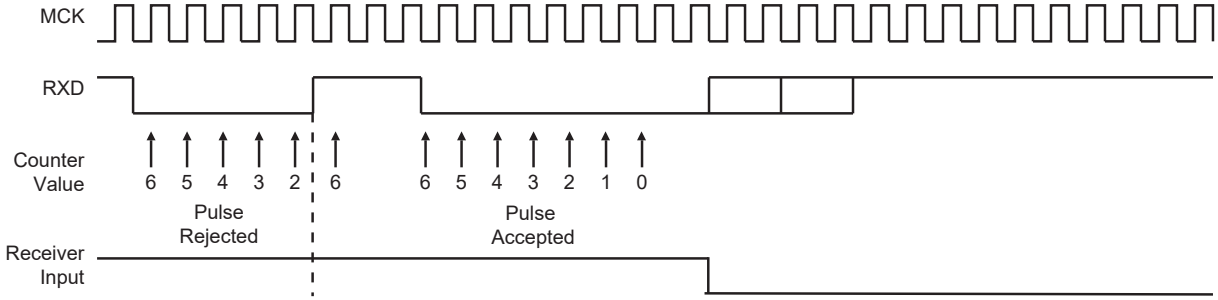
| Peripheral Clock | Baud Rate (bit/s) | CD | Baud Rate Error | Pulse Time (µs) |
|---|---|---|---|---|
| 3,686,400 | 115,200 | 2 | 0.00% | 1.63 |
| 20,000,000 | 115,200 | 11 | 1.38% | 1.63 |
| 32,768,000 | 115,200 | 18 | 1.25% | 1.63 |
| 40,000,000 | 115,200 | 22 | 1.38% | 1.63 |
| 3,686,400 | 57,600 | 4 | 0.00% | 3.26 |
| 20,000,000 | 57,600 | 22 | 1.38% | 3.26 |
| 32,768,000 | 57,600 | 36 | 1.25% | 3.26 |
| 40,000,000 | 57,600 | 43 | 0.93% | 3.26 |
| 3,686,400 | 38,400 | 6 | 0.00% | 4.88 |
| 20,000,000 | 38,400 | 33 | 1.38% | 4.88 |
| 32,768,000 | 38,400 | 53 | 0.63% | 4.88 |
| 40,000,000 | 38,400 | 65 | 0.16% | 4.88 |
| 3,686,400 | 19,200 | 12 | 0.00% | 9.77 |
| 20,000,000 | 19,200 | 65 | 0.16% | 9.77 |
| 32,768,000 | 19,200 | 107 | 0.31% | 9.77 |
| 40,000,000 | 19,200 | 130 | 0.16% | 9.77 |
| 3,686,400 | 9,600 | 24 | 0.00% | 19.53 |
| 20,000,000 | 9,600 | 130 | 0.16% | 19.53 |
| 32,768,000 | 9,600 | 213 | 0.16% | 19.53 |
| 40,000,000 | 9,600 | 260 | 0.16% | 19.53 |
| 3,686,400 | 2,400 | 96 | 0.00% | 78.13 |
| 20,000,000 | 2,400 | 521 | 0.03% | 78.13 |
| 32,768,000 | 2,400 | 853 | 0.04% | 78.13 |

#### 46.7.5.3 IrDA Demodulator

The demodulator is based on the IrDA Receive filter comprised of an 8-bit down counter which is loaded with the value programmed in FLEX_US_IF. When a falling edge is detected on the RXD pin, the Filter Counter starts counting down at the peripheral clock speed. If a rising edge is detected on the RXD pin, the counter stops and is reloaded with FLEX_US_IF. If no rising edge is detected when the counter reaches 0, the input of the receiver is driven low during one bit time.

The following figure illustrates the operations of the IrDA demodulator.

**Figure 46-35. IrDA Demodulator Operations**




The programmed value in the FLEX_US_IF register must always meet the following criteria:

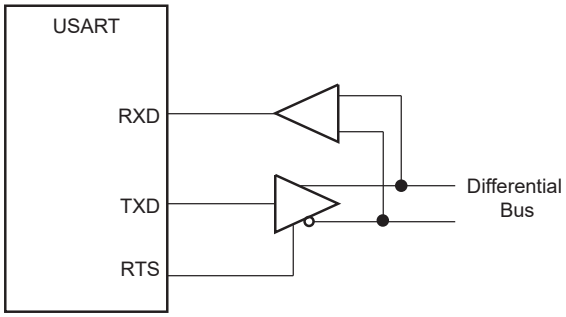
$t_{peripheral\ clock} \times (IRDA_FILTER + 3) < 1.41\ \mu s$

As the IrDA mode uses the same logic as the ISO7816, note that the FLEX_US_FIDI.FI_DI_RATIO field must be set to a value higher than 0 to make sure IrDA communications operate correctly.

### 46.7.6 RS485 Mode

The USART features the RS485 mode to enable line driver control. While operating in RS485 mode, the USART behaves as though in Asynchronous or Synchronous mode and configuration of all the parameters is possible. The difference is that the RTS pin is driven high when the transmitter is operating. The behavior of the RTS pin is controlled by the TXEMPTY bit. A typical connection of the USART to an RS485 bus is shown in the following figure.

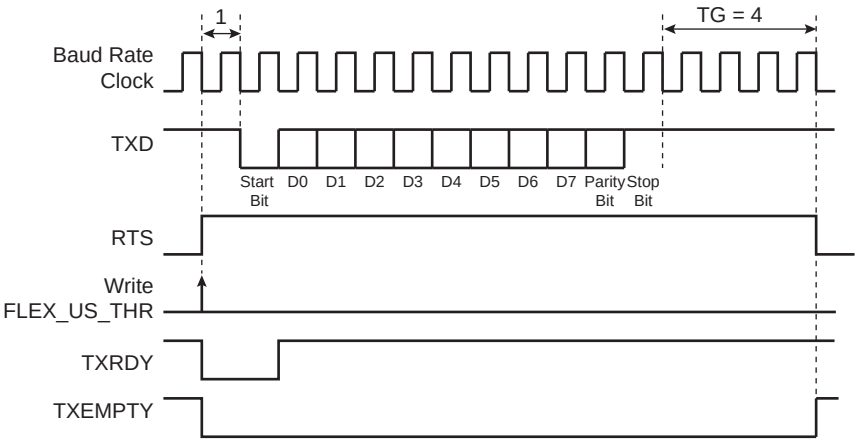
**Figure 46-36. Typical Connection to an RS485 Bus**




The USART is set in RS485 mode by writing the value 0x1 to the FLEX_US_MR.USART_MODE field.

The RTS pin is at a level inverse to the TXEMPTY bit. Significantly, the RTS pin remains high when a timeguard is programmed, so that the line can remain driven after the last character completion. The following figure gives an example of the RTS waveform during a character transmission when the timeguard is enabled.

**Figure 46-37. Example of RTS Drive with Timeguard**




### 46.7.7 USART Comparison Function on Received Character

The CMP flag in FLEX_US_CSR is set when the received character matches the conditions programmed in FLEX_US_CMPR. The CMP flag is set as soon as FLEX_US_RHR is loaded with the new received character. The CMP flag is cleared by writing a one to FLEX_US_CR.RSTSTA.

FLEX_US_CMPR can be programmed to provide different comparison methods:

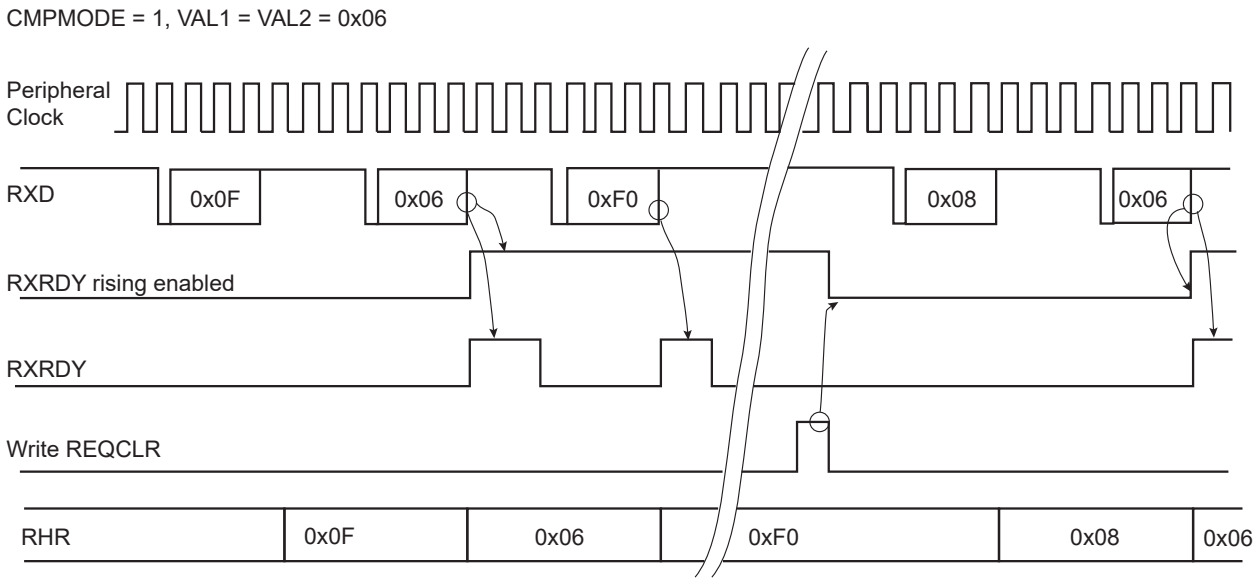
- If VAL1 equals VAL2, then the comparison is performed on a single value and the flag is set to 1 if the received character equals VAL1.
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 sets the CMP flag.
- If VAL1 is strictly higher than VAL2, then the flag CMP is set to 1 if any received character equals VAL1 or VAL2.

When the FLEX_US_CMPR.CMPMODE bit is set to FLAG_ONLY (value 0), all received data are loaded in FLEX_US_RHR and the CMP flag provides the status of the comparison result.

By programming the START_CONDITION.CMPMODE bit (value 1), the comparison function result triggers the start of the loading of FLEX_US_RHR (see the following figure). The trigger condition exists as soon as the received character value matches the condition defined by the programming of VAL1, VAL2 and CMPPAR in FLEX_US_CMPR. The comparison trigger event is restarted by writing a 1 to the FLEX_US_CR.REQCLR bit.

The value programmed in the VAL1 and VAL2 fields must not exceed the maximum value of the received character (see CHRL field in register 46.10.6 FLEX_US_MR).

**Figure 46-38. Receive Holding Register Management**




### 46.7.8 SPI Mode

The Serial Peripheral Interface (SPI) mode is a synchronous serial data link that provides communication with external devices in Master or Slave mode. It also enables communication between processors if an external processor is connected to the system.

The Serial Peripheral Interface is essentially a shift register that serially transmits data bits to other SPIs. During a data transfer, one SPI system acts as the “master” which controls the data flow, while the other devices act as “slaves" which have data shifted into and out by the master. Different CPUs can take turns being masters and one master may simultaneously shift data into multiple slaves. (Multiple master protocol is the opposite of single master protocol, where one CPU is always the master while all of the others are always slaves.) However, only one slave may drive its output to write data back to the master at any given time.

A slave device is selected when its NSS signal is asserted by the master. The USART in SPI Master mode can address only one SPI slave because it can generate only one NSS signal.

The SPI system consists of two data lines and two control lines:

- Master Out Slave In (MOSI): This data line supplies the output data from the master shifted into the input of the slave.
- Master In Slave Out (MISO): This data line supplies the output data from a slave to the input of the master.
- Serial Clock (SCK): This control line is driven by the master and regulates the flow of the data bits. The master may transmit data at a variety of bit rates. The SCK line cycles once for each bit that is transmitted.
- Slave Select (NSS): This control line allows the master to select or deselect the slave.

#### 46.7.8.1 Modes of Operation

The USART can operate in SPI Master mode or in SPI Slave mode.

Operation in SPI Master mode is programmed by writing 0xE to the FLEX_US_MR.USART_MODE field. In this case, the SPI lines must be connected as described below:

- The MOSI line is driven by the output pin TXD

- The MISO line drives the input pin RXD
- The SCK line is driven by the output pin SCK
- The NSS line is driven by the output pin RTS

Operation in SPI Slave mode is programmed by writing to 0xF the FLEX_US_MR.USART_MODE field. In this case, the SPI lines must be connected as described below:

- The MOSI line drives the input pin RXD
- The MISO line is driven by the output pin TXD
- The SCK line drives the input pin SCK
- The NSS line drives the input pin CTS

In order to avoid an unpredictable behavior, any change of the SPI mode must be followed by a software reset of the transmitter and of the receiver (except the initial configuration after a hardware reset). (See section Receiver and Transmitter Control.)

#### 46.7.8.2 Bit Rate

In SPI mode, the bit rate generator operates in the same way as in USART Synchronous mode. See section 46.7.1.3 Baud Rate in Synchronous Mode or SPI Mode. However, some restrictions apply:

In SPI Master mode:

- The external clock SCK must not be selected (USCLKS ≠ 0x3), and the FLEX_US_MR.CLKO bit must be set in order to generate correctly the serial clock on the SCK pin.
- To ensure a correct behavior of the receiver and the transmitter, the value programmed in CD must be ≥ 6.
- If the divided peripheral clock is selected, the value programmed in CD must be even to ensure a 50:50 mark/space ratio on the SCK pin; this value can be odd if the peripheral clock is selected.

In SPI Slave mode:

- The external clock (SCK) selection is forced regardless of the value of the FLEX_US_MR.USCLKS field. Likewise, the value written in FLEX_US_BRGR has no effect, because the clock is provided directly by the signal on the USART SCK pin.
- To ensure a correct behavior of the receiver and the transmitter, the external clock (SCK) frequency must be at least six times lower than the system clock.

#### 46.7.8.3 Data Transfer

Up to nine data bits are successively shifted out on the TXD pin at each rising or falling edge (depending of CPOL and CPHA) of the programmed serial clock. There is no Start bit, no Parity bit and no Stop bit.

The number of data bits is selected by the CHRL field and the MODE9 bit in FLEX_US_MR. The nine bits are selected by setting the MODE9 bit regardless of the CHRL field. The MSB data bit is always sent first in SPI mode (Master or Slave).

Four combinations of polarity and phase are available for data transfers. The clock polarity is programmed with the FLEX_US_MR.CPOL bit. The clock phase is programmed with the CPHA bit. These two parameters determine the edges of the clock signal upon which data are driven and sampled. Each of the two parameters has two possible states, resulting in four possible combinations that are incompatible with one another. Thus, a master/slave pair must use the same parameter pair values to communicate. If multiple slaves are used and fixed in different configurations, the master must reconfigure itself each time it needs to communicate with a different slave.

**Table 46-12. SPI Bus Protocol Mode**

| SPI Bus Protocol Mode | CPOL | CPHA |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 0 | 0 |
| 2 | 1 | 1 |
| 3 | 1 | 0 |

**Figure 46-39. SPI Transfer Format (CPHA = 1, 8 bits per transfer)**

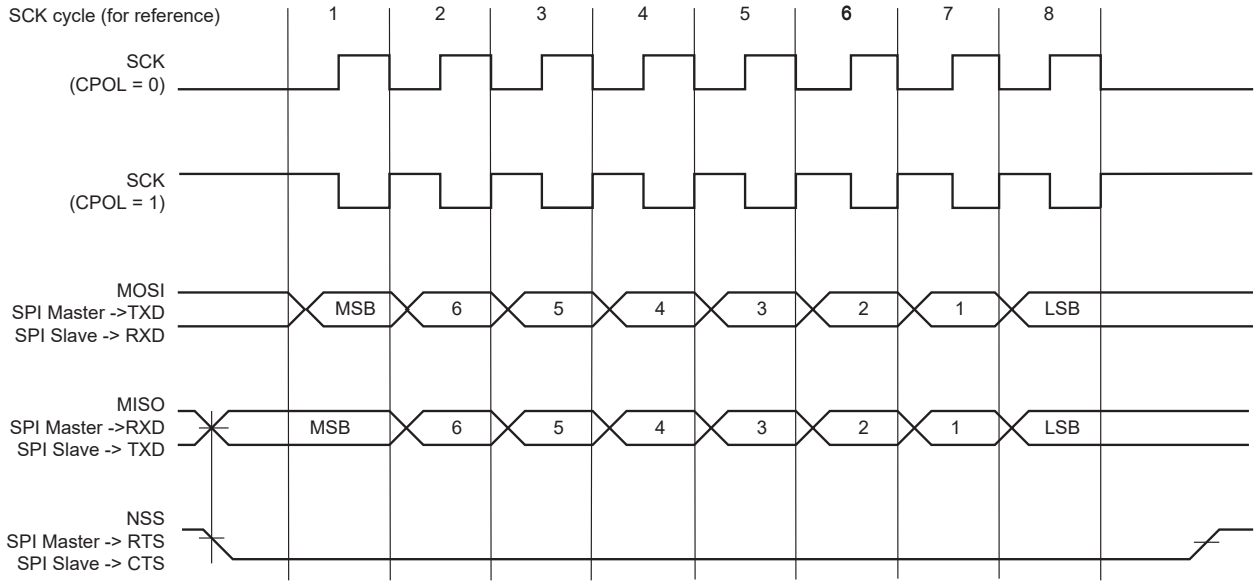
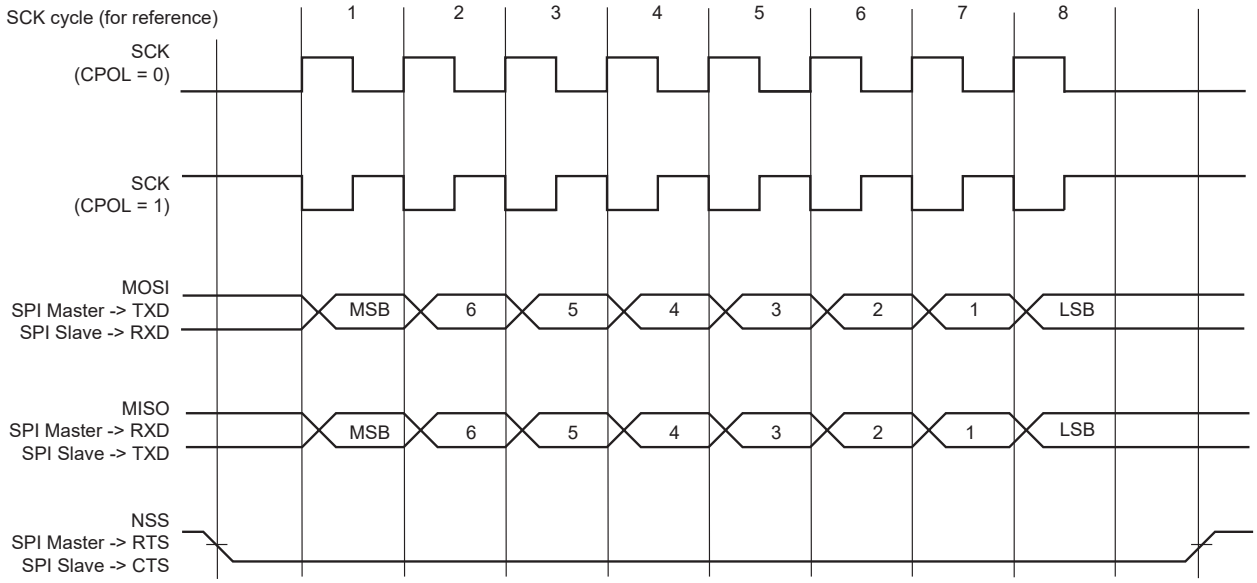



**Figure 46-40. SPI Transfer Format (CPHA = 0, 8 bits per transfer)**




#### 46.7.8.4 Receiver and Transmitter Control

See section Receiver and Transmitter Control.

#### 46.7.8.5 Character Transmission

The characters are sent by writing in the Transmit Holding Register (FLEX_US_THR). An additional condition for transmitting a character can be added when the USART is configured in SPI Master mode. In the “USART Mode Register (SPI_MODE)” (FLEX_US_MR), the value configured on the WRDBT bit can prevent any character transmission (even if FLEX_US_THR has been written) while the receiver side is not ready (character not read). When WRDBT = 0, the character is transmitted whatever the receiver status. If WRDBT = 1, the transmitter waits for the Receive Holding Register (FLEX_US_RHR) to be read before transmitting the character (RXRDY flag cleared), thus preventing any overflow (character loss) on the receiver side.

The chip select line is deasserted for a period equivalent to 3 bits between the transmission of two data.

The transmitter reports two status bits in FLEX_US_CSR: TXRDY (Transmitter Ready), which indicates that FLEX_US_THR is empty and TXEMPTY, which indicates that all the characters written in FLEX_US_THR have been processed. When the current character processing is completed, the last character written in FLEX_US_THR is transferred into the shift register of the transmitter and FLEX_US_THR is emptied, and thus TXRDY rises.

Both the TXRDY and the TXEMPTY bits are low when the transmitter is disabled. Writing a character in FLEX_US_THR while TXRDY is low has no effect and the written character is lost.

If the USART is in SPI Slave mode and if a character must be sent while FLEX_US_THR is empty, the UNRE (Underrun Error) bit is set. The TXD transmission line stays at high level during all this time. The UNRE bit is cleared by writing a one to the FLEX_US_CR.RSTSTA bit.

In SPI Master mode, the slave select line (NSS) is asserted at low level one $t_{bit}$ ($t_{bit}$ being the nominal time required to transmit a bit) before the transmission of the MSB bit and released at high level one $t_{bit}$ after the transmission of the LSB bit. So, the slave select line (NSS) is always released between each character transmission and a minimum delay of three $t_{bit}$ always inserted. However, in order to address slave devices supporting the CSAAT mode (Chip Select Active After Transfer), the slave select line (NSS) can be forced at low level by writing a one to the FLEX_US_CR.RTSEN bit. The slave select line (NSS) can be released at high level only by writing a one to the FLEX_US_CR.RTSDIS bit (for example, when all data have been transferred to the slave device).

In SPI Slave mode, the transmitter does not require a falling edge of the slave select line (NSS) to initiate a character transmission but only a low level. However, this low level must be present on the slave select line (NSS) at least one $t_{bit}$ before the first serial clock cycle corresponding to the MSB bit.

#### 46.7.8.6 Character Reception

When a character reception is completed, it is transferred to the Receive Holding Register (FLEX_US_RHR) and the RXRDY bit in the Status Register (FLEX_US_CSR) rises. If a character is completed while RXRDY is set, the OVRE (Overrun Error) bit is set. The last character is transferred into FLEX_US_RHR and overwrites the previous one. The OVRE bit is cleared by writing a one to the FLEX_US_CR.RSTSTA bit.

To ensure correct behavior of the receiver in SPI Slave mode, the master device sending the frame must ensure a minimum delay of one $t_{bit}$ between each character transmission. The receiver does not require a falling edge of the slave select line (NSS) to initiate a character reception but only a low level. However, this low level must be present on the slave select line (NSS) at least one $t_{bit}$ before the first serial clock cycle corresponding to the MSB bit.

#### 46.7.8.7 Receiver Timeout

Because the receiver bit rate clock is active only during data transfers in SPI mode, a receiver timeout is impossible in this mode, whatever the timeout value is in field FLEX_US_RTOR.TO.

### 46.7.9 LIN Mode

The LIN mode provides master node and slave node connectivity on a LIN bus.

The LIN (Local Interconnect Network) is a serial communication protocol which efficiently supports the control of mechatronic nodes in distributed automotive applications.

The main properties of the LIN bus are:

- Single master/multiple slaves concept
- Low-cost silicon implementation based on common UART/SCI interface hardware, an equivalent in software, or as a pure state machine.
- Self synchronization without quartz or ceramic resonator in the slave nodes
- Deterministic signal transmission
- Low cost single-wire implementation
- Speed up to 20 kbit/s

LIN provides cost efficient bus communication where the bandwidth and versatility of CAN are not required.

The LIN mode enables processing LIN frames with a minimum of action from the microprocessor.

#### 46.7.9.1 Modes of Operation

The USART can act either as a LIN master node or as a LIN slave node.

The node configuration is chosen by setting the USART_MODE field in the USART Mode Register (FLEX_US_MR):

- LIN master node (USART_MODE = 0xA)
- LIN slave node (USART_MODE = 0xB)

In order to avoid unpredictable behavior, any change of the LIN node configuration must be followed by a software reset of the transmitter and of the receiver (except the initial node configuration after a hardware reset). (See section 46.7.2 Receiver and Transmitter Control.)

#### 46.7.9.2 Baud Rate Configuration

See section Baud Rate in Asynchronous Mode.

- LIN master node: The baud rate is configured in FLEX_US_BRGR.
- LIN slave node: The initial baud rate is configured in FLEX_US_BRGR. This configuration is automatically copied in the LIN Baud Rate Register (FLEX_US_LINBRR) when writing FLEX_US_BRGR. After the synchronization procedure, the baud rate is updated in FLEX_US_LINBRR.

#### 46.7.9.3 Receiver and Transmitter Control

See section 46.7.2 Receiver and Transmitter Control.

#### 46.7.9.4 Character Transmission

See section Transmitter Operations.

#### 46.7.9.5 Character Reception

See section Receiver Operations.

#### 46.7.9.6 Header Transmission (Master Node Configuration)

All the LIN Frames start with a header which is sent by the master node and consists of a Synch Break Field, Synch Field and Identifier Field.

So in master node configuration, the frame handling starts with the sending of the header.

The header is transmitted as soon as the identifier is written in the LIN Identifier Register (FLEX_US_LINIR). At this moment the flag TXRDY falls.

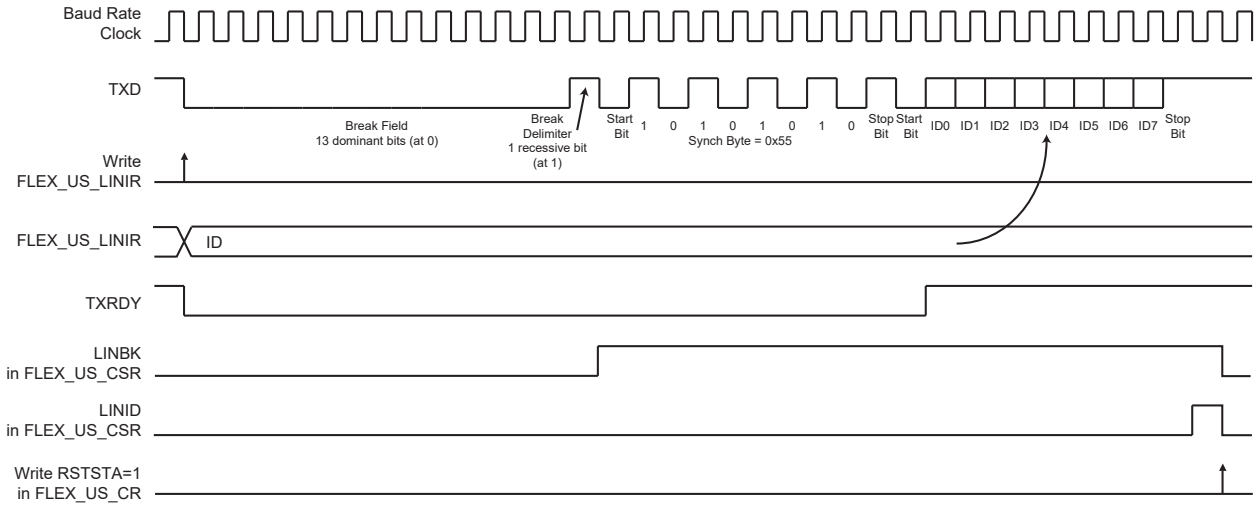
The Break Field, the Synch Field and the Identifier Field are sent automatically one after the other.

The Break Field consists of 13 dominant bits and 1 recessive bit, the Synch Field is the character 0x55 and the Identifier corresponds to the character written in the LIN Identifier Register (FLEX_US_LINIR). The Identifier parity bits can be automatically computed and sent (see section Identifier Parity).

The flag TXRDY rises when the identifier character is transferred into the shift register of the transmitter.

As soon as the Synch Break Field is transmitted, the FLEX_US_CSR.LINBK flag bit is set. Likewise, as soon as the Identifier Field is sent, the FLEX_US_CSR.LINID flag bit is set. These flags are reset by writing a one to the FLEX_US_CR.RSTSTA bit.

**Figure 46-41. Header Transmission**




### 46.7.9.7 Header Reception (Slave Node Configuration)

All the LIN frames start with a header which is sent by the master node and consists of a Synch Break Field, Synch Field and Identifier Field.

In slave node configuration, the frame handling starts with the reception of the header.

The USART uses a break detection threshold of 11 nominal bit times at the actual baud rate. At any time, if 11 consecutive recessive bits are detected on the bus, the USART detects a Break Field. As long as a Break Field has not been detected, the USART stays idle and the received data are not taken in account.

When a Break Field has been detected, the FLEX_US_CSR.LINBK flag is set and the USART expects the Synch Field character to be 0x55. This field is used to update the actual baud rate in order to remain synchronized (see section Slave Node Synchronization). If the received Synch character is not 0x55, an Inconsistent Synch Field error is generated (see section LIN Errors).

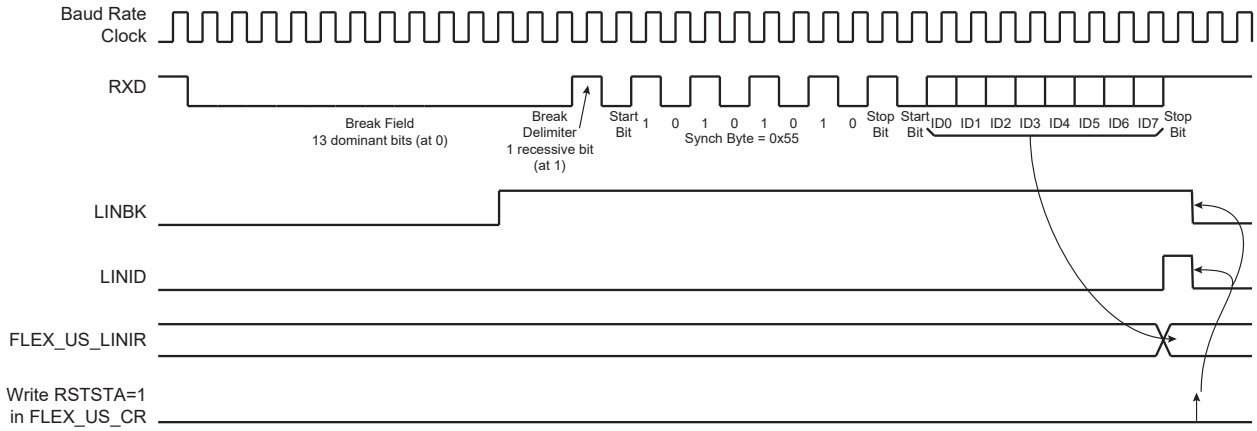
After receiving the Synch Field, the USART expects to receive the Identifier Field.

When the Identifier Field has been received, the FLEX_US_CSR.LINID flag bit is set. At this moment, the IDCHR field in the LIN Identifier Register (FLEX_US_LINIR) is updated with the received character. The Identifier parity bits can be automatically computed and checked (see section Identifier Parity).

If the header is not entirely received within the time given by the maximum length of the header $t_{Header_Maximum}$, the FLEX_US_CSR.LINHTE error flag bit is set.

The flag bits LINID, LINBK and LINHTE are reset by writing a one to the FLEX_US_CR.RSTSTA bit.

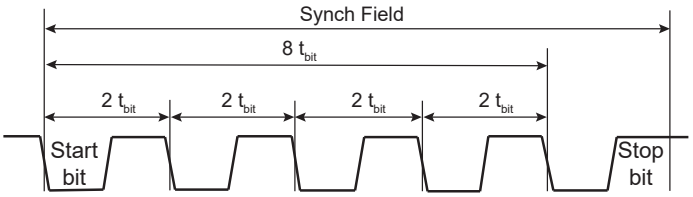
**Figure 46-42. Header Reception**




### 46.7.9.8 Slave Node Synchronization

The synchronization is done only in slave node configuration. The procedure is based on time measurement between the falling edges of the Synch Field. The falling edges are available in distances of 2, 4, 6 and 8 bit times.

**Figure 46-43. Synch Field**




The time measurement is made by a 19-bit counter driven by the sampling clock (see section Baud Rate Generator).

When the start bit of the Synch Field is detected, the counter is reset. Then during the next eight $t_{bit}$ of the Synch Field, the counter is incremented. At the end of these eight $t_{bit}$, the counter is stopped. At this moment, the 16 most significant bits of the counter (value divided by 8) give the new clock divider (LINCD) and the 3 least significant bits of this value (the remainder) give the new fractional part (LINFP).

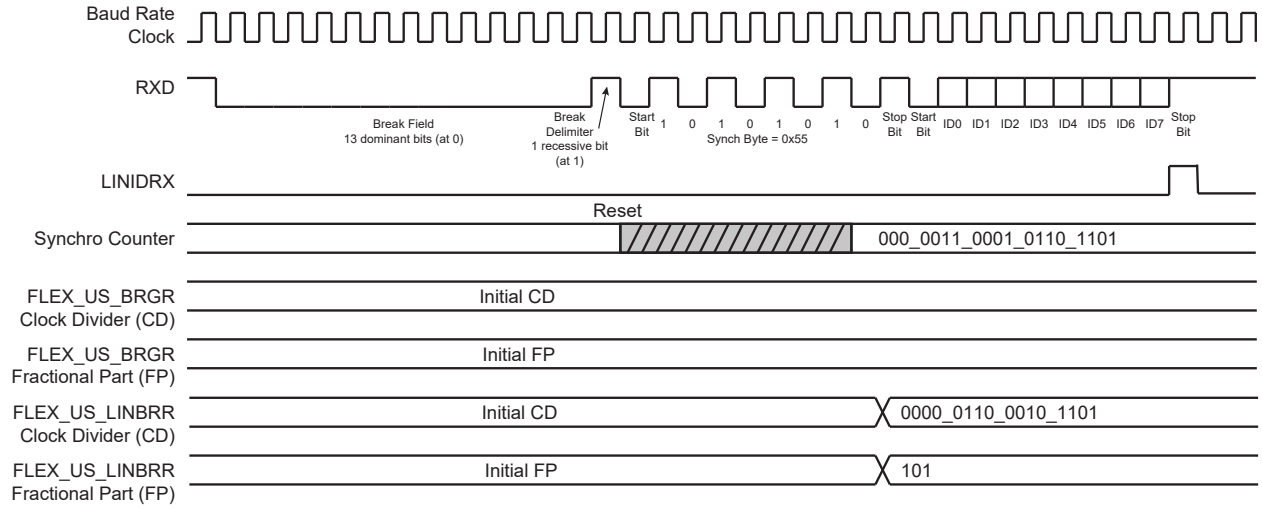
Once the Synch Field has been entirely received, the clock divider (LINCD) and the fractional part (LINFP) are updated in the LIN Baud Rate Register (FLEX_US_LINBRR) with the computed values, if the Synchronization is not disabled by the SYNCDIS bit in the LIN Mode Register (FLEX_US_LINMR).

After reception of the Synch Field:

- If it appears that the computed baud rate deviation compared to the initial baud rate is superior to the maximum tolerance FTol_Unsynch (±15%), then the clock divider (LINCD) and the fractional part (LINFP) are not updated, and the FLEX_US_CSR.LINSTE error flag bit is set.
- If it appears that the sampled Synch character is not equal to 0x55, then the clock divider (LINCD) and the fractional part (LINFP) are not updated, and the FLEX_US_CSR.LINISFE error flag bit is set.

Flags LINSTE and LINISFE are reset by writing a one to the FLEX_US_CR.RSTSTA bit.

**Figure 46-44. Slave Node Synchronization**




The synchronization accuracy depends on several parameters:

- The nominal clock frequency ($f_{Nom}$) (the theoretical slave node clock frequency)
- The baud rate
- The oversampling (OVER = 0 => 16X or OVER = 1 => 8X)

The following formula is used to compute the deviation of the slave bit rate relative to the master bit rate after synchronization ($f_{SLAVE}$ is the real slave node clock frequency).

$$\text{Baud rate deviation} = \left(100 \times \frac{[\alpha \times 8 \times (2 - \text{Over}) + \beta] \times \text{Baud rate}}{8 \times f_{SLAVE}}\right)\%$$

$$\text{Baud rate deviation} = \left(100 \times \frac{[\alpha \times 8 \times (2 - \text{Over}) + \beta] \times \text{Baud rate}}{8 \times \left(\frac{f_{TOL_UNSYNCH}}{100}\right) \times f_{Nom}}\right)\%$$

$-0.5 \leq \alpha \leq +0.5 \quad -1 < \beta < +1$

$f_{TOL_UNSYNCH}$ is the deviation of the real slave node clock from the nominal clock frequency. The LIN Standard imposes that it must not exceed ±15%. The LIN Standard imposes also that for communication between two nodes, their bit rate must not differ by more than ±2%. This means that the baud rate deviation must not exceed ±1%.

Therefore, a minimum value for the nominal clock frequency can be computed as follows:

$$f_{Nom}(\text{min}) = \left(100 \times \frac{[0.5 \times 8 \times (2 - \text{Over}) + 1] \times \text{Baud rate}}{8 \times \left(\frac{-15}{100} + 1\right) \times 1\%}\right)\text{Hz}$$

Examples:

- Baud rate = 20 kbit/s, OVER = 0 (Oversampling 16X) => $f_{Nom}$(min) = 2.64 MHz
- Baud rate = 20 kbit/s, OVER = 1 (Oversampling 8X) => $f_{Nom}$(min) = 1.47 MHz
- Baud rate = 1 kbit/s, OVER = 0 (Oversampling 16X) => $f_{Nom}$(min) = 132 kHz
- Baud rate = 1 kbit/s, OVER = 1 (Oversampling 8X) => $f_{Nom}$(min) = 74 kHz

#### 46.7.9.9 Identifier Parity

A protected identifier consists of two subfields: the identifier and the identifier parity. Bits 0 to 5 are assigned to the identifier, and bits 6 and 7 are assigned to the parity.

The USART interface can generate/check these parity bits, but this feature can also be disabled. The user can choose between two modes via the FLEX_US_LINMR.PARDIS bit:

- PARDIS = 0:
  - During header transmission, the parity bits are computed and sent with the six least significant bits of the IDCHR field of the LIN Identifier Register (FLEX_US_LINIR). Bits 6 and 7 of this register are discarded.
  - During header reception, the parity bits of the identifier are checked. If the parity bits are wrong, an Identifier Parity error occurs (see section Parity). Only the six least significant bits of the IDCHR field are updated with the received Identifier. Bits 6 and 7 are stuck to 0.
- PARDIS = 1:
  - During header transmission, all the bits of the IDCHR field of the LIN Identifier Register (FLEX_US_LINIR) are sent on the bus.
  - During header reception, all the bits of the IDCHR field are updated with the received Identifier.

#### 46.7.9.10 Node Action

Depending on the identifier, the node is affected—or not—by the LIN response. Consequently, after sending or receiving the identifier, the USART must be configured. There are three possible configurations:

- PUBLISH: the node sends the response.
- SUBSCRIBE: the node receives the response.
- IGNORE: the node is not concerned by the response, it does not send and does not receive the response.

This configuration is made by the LIN Node Action (NACT) field in USART LIN Mode Register (FLEX_US_LINMR).

Example: a LIN cluster that contains a master and two slaves:

- Data transfer from the master to slave 1 and to slave 2:

NACT(master) = PUBLISH

NACT(slave 1) = SUBSCRIBE

NACT(slave 2) = SUBSCRIBE

- Data transfer from the master to slave 1 only:

NACT(master) = PUBLISH

NACT(slave 1) = SUBSCRIBE

NACT(slave 2) = IGNORE

- Data transfer from slave 1 to the master:

NACT(master) = SUBSCRIBE

NACT(slave 1) = PUBLISH

NACT(slave 2) = IGNORE

- Data transfer from slave 1 to slave 2:

NACT(master) = IGNORE

NACT(slave 1) = PUBLISH

NACT(slave 2) = SUBSCRIBE

- Data transfer from slave 2 to the master and to slave 1:

NACT(master) = SUBSCRIBE

NACT(slave 1) = SUBSCRIBE

NACT(slave 2) = PUBLISH

#### 46.7.9.11 Response Data Length

The LIN response data length is the number of data fields (bytes) of the response excluding the checksum.

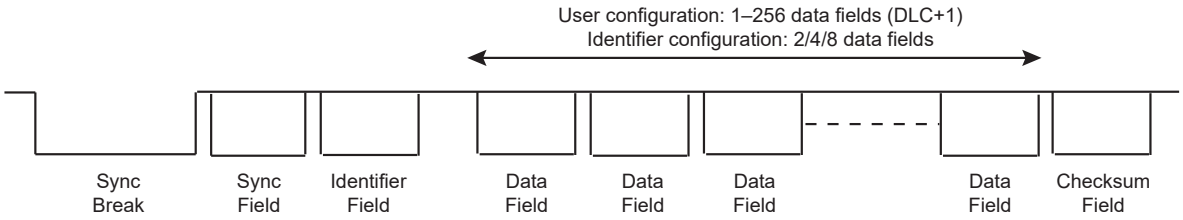
The response data length can either be configured by the user or be defined automatically by bits 4 and 5 of the Identifier (compatibility to LIN Specification 1.1). The user can choose between these two modes by the FLEX_US_LINMR.DLM bit:

- DLM = 0: The response data length is configured by the user via the FLEX_US_LINMR.DLC field. The response data length is equal to (DLC + 1) bytes. DLC can be programmed from 0 to 255, so the response can contain from 1 data byte up to 256 data bytes.
- DLM = 1: The response data length is defined by the Identifier (IDCHR in FLEX_US_LINIR) according to the table below. The FLEX_US_LINMR.DLC field is discarded. The response can contain 2 or 4 or 8 data bytes.

**Table 46-13. Response Data Length if DLM = 1**

| IDCHR[5] | IDCHR[4] | Response Data Length (bytes) |
|---|---|---|
| 0 | 0 | 2 |
| 0 | 1 | 2 |
| 1 | 0 | 4 |
| 1 | 1 | 8 |

**Figure 46-45. Response Data Length**




#### 46.7.9.12 Checksum

The last field of a frame is the checksum. The checksum contains the inverted 8- bit sum with carry, over all data bytes or all data bytes and the protected identifier. Checksum calculation over the data bytes only is called classic checksum and it is used for communication with LIN 1.3 slaves. Checksum calculation over the data bytes and the protected identifier byte is called enhanced checksum and it is used for communication with LIN 2.0 slaves.

The USART can be configured to:

- Send/Check an Enhanced checksum automatically (CHKDIS = 0 & CHKTYP = 0)
- Send/Check a Classic checksum automatically (CHKDIS = 0 & CHKTYP = 1)
- Not send/check a checksum (CHKDIS = 1)

This configuration is made by the Checksum Type (CHKTYP) and Checksum Disable (CHKDIS) bits of FLEX_US_LINMR.

If the checksum feature is disabled, the user can send it manually all the same, by considering the checksum as a normal data byte and by adding 1 to the response data length (see section Response Data Length).

#### 46.7.9.13 Frame Slot Mode

This mode is useful only for master nodes. It respects the following rule: each frame slot shall be longer than or equal to $t_{Frame_Maximum}$.

If the Frame Slot mode is enabled (FSDIS = 0) and a frame transfer has been completed, the TXRDY flag is set again only after $t_{Frame_Maximum}$ delay, from the start of frame. So the master node cannot send a new header if the frame slot duration of the previous frame is inferior to $t_{Frame_Maximum}$.

If the Frame Slot mode is disabled (FSDIS = 1) and a frame transfer has been completed, the TXRDY flag is set again immediately.

The $t_{Frame_Maximum}$ is calculated as follows:

If the Checksum is sent (CHKDIS = 0):

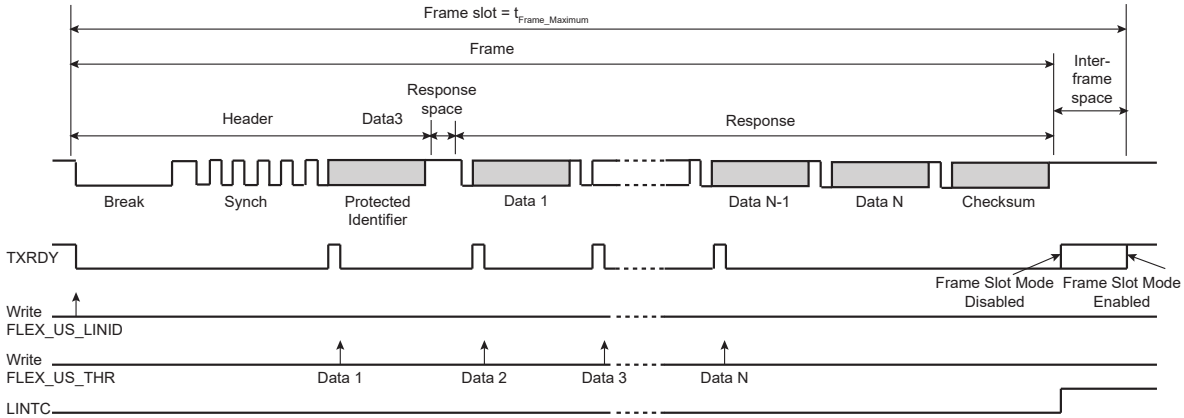
- $t_{Header_Nominal} = 34 \times t_{bit}$
- $t_{Response_Nominal} = 10 \times (NData + 1) \times t_{bit}$
- $t_{Frame_Maximum} = 1.4 \times (t_{Header_Nominal} + t_{Response_Nominal} + 1)$(1)
- $t_{Frame_Maximum} = 1.4 \times (34 + 10 \times (DLC + 1 + 1) + 1) \times t_{bit}$
- $t_{Frame_Maximum} = (77 + 14 \times DLC) \times t_{bit}$

If the Checksum is not sent (CHKDIS = 1):

- $t_{Header_Nominal} = 34 \times t_{bit}$
- $t_{Response_Nominal} = 10 \times NData \times t_{bit}$
- $t_{Frame_Maximum} = 1.4 \times (t_{Header_Nominal} + t_{Response_Nominal} + 1)$(1)
- $t_{Frame_Maximum} = 1.4 \times (34 + 10 \times (DLC + 1) + 1) \times t_{bit}$
- $t_{Frame_Maximum} = (63 + 14 \times DLC) \times t_{bit}$

Note: 1. The term “+1” leads to an integer result for $t_{Frame_Maximum}$ (LIN Specification 1.3).

**Figure 46-46. Frame Slot Mode**




### 46.7.9.14 LIN Errors

#### 46.7.9.14.1 Bit Error

This error is generated in master of slave node configuration, when the USART is transmitting and if the transmitted value on the Tx line is different from the value sampled on the Rx line. If a bit error is detected, the transmission is aborted at the next byte border.

This error is reported by the FLEX_US_CSR.LINBE flag.

#### 46.7.9.14.2 Inconsistent Synch Field Error

This error is generated in slave node configuration, if the Synch Field character received is other than 0x55.

This error is reported by the FLEX_US_CSR.LINISFE flag.

#### 46.7.9.14.3 Identifier Parity Error

This error is generated in slave node configuration, if the parity of the identifier is wrong. This error can be generated only if the parity feature is enabled (PARDIS = 0).

This error is reported by the FLEX_US_CSR.LINIPE flag.

#### 46.7.9.14.4 Checksum Error

This error is generated in master of slave node configuration, if the received checksum is wrong. This flag can be set to 1 only if the checksum feature is enabled (CHKDIS = 0).

This error is reported by the FLEX_US_CSR.LINCE flag.

#### 46.7.9.14.5 Slave Not Responding Error

This error is generated in master of slave node configuration, when the USART expects a response from another node (NACT = SUBSCRIBE) but no valid message appears on the bus within the time given by the maximum length of the message frame, $t_{Frame_Maximum}$ (see section Frame Slot Mode). This error is disabled if the USART does not expect any message (NACT = PUBLISH or NACT = IGNORE).

This error is reported by the FLEX_US_CSR.LINSNRE.

#### 46.7.9.14.6 Synch Tolerance Error

This error is generated in slave node configuration if, after the clock synchronization procedure, it appears that the computed baud rate deviation compared to the initial baud rate is superior to the maximum tolerance FTol_Unsynch (±15%).

This error is reported by the FLEX_US_CSR.LINSTE flag.

#### 46.7.9.14.7 Header Timeout Error

This error is generated in slave node configuration, if the Header is not entirely received within the time given by the maximum length of the Header, $t_{Header_Maximum}$.

This error is reported by the FLEX_US_CSR.LINHTE flag.

### 46.7.9.15 LIN Frame Handling

#### 46.7.9.15.1 Master Node Configuration

- Write FLEX_US_CR.TXEN and FLEX_US_CR.RXEN to enable both the transmitter and the receiver.
- Write FLEX_US_MR.USART_MODE to select the LIN mode and the master node configuration.
- Write FLEX_US_BRGR.CD and FLEX_US_BRGR.FP to configure the baud rate.
- Write NACT, PARDIS, CHKDIS, CHKTYPE, DLCM, FSDIS and DLC in FLEX_US_LINMR to configure the frame transfer.
- Check that FLEX_US_CSR. TXRDY is set to 1.
- Write FLEX_US_LINIR.IDCHR to send the header.

What comes next depends on the NACT configuration:

- Case 1: NACT = PUBLISH, the USART sends the response.
  - Wait until FLEX_US_CSR. TXRDY rises.
  - Write FLEX_US_THR.TCHR to send a byte.
  - If all the data have not been written, repeat the two previous steps.
  - Wait until FLEX_US_CSR.LINTC rises.
  - Check the LIN errors.
- Case 2: NACT = SUBSCRIBE, the USART receives the response.
  - Wait until FLEX_US_CSR.RXRDY rises.
  - Read FLEX_US_RHR.RCHR.
  - If all the data have not been read, repeat the two previous steps.
  - Wait until FLEX_US_CSR.LINTC rises.
  - Check the LIN errors.
- Case 3: NACT = IGNORE, the USART is not concerned by the response.
  - Wait until FLEX_US_CSR.LINTC rises.
  - Check the LIN errors.

**Figure 46-47. Master Node Configuration, NACT = PUBLISH**

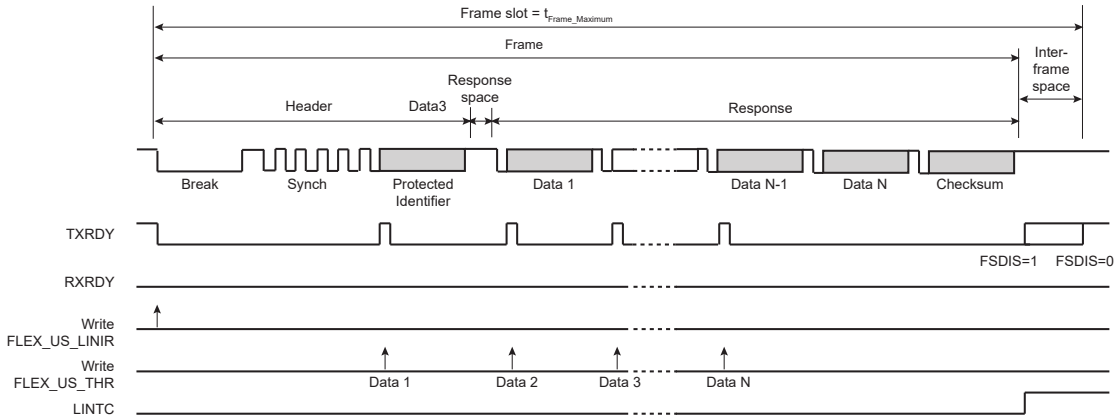


**Figure 46-48. Master Node Configuration, NACT = SUBSCRIBE**

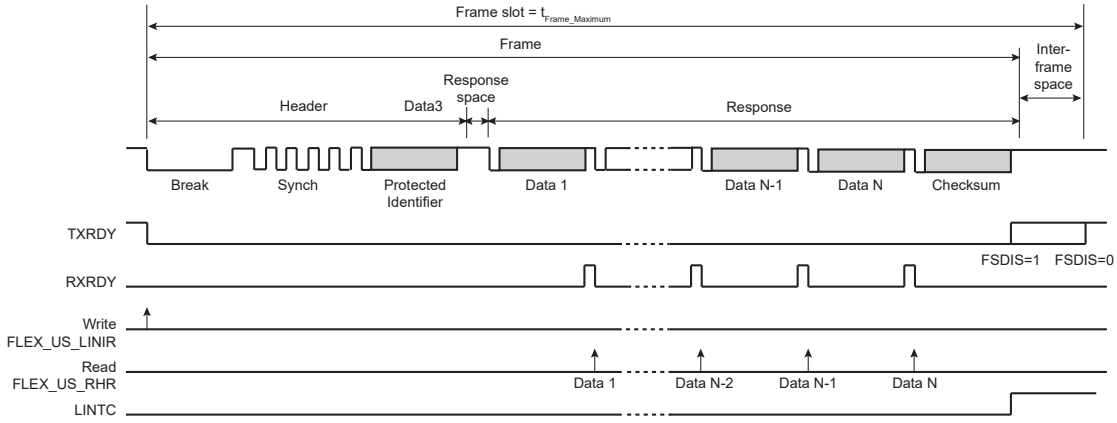
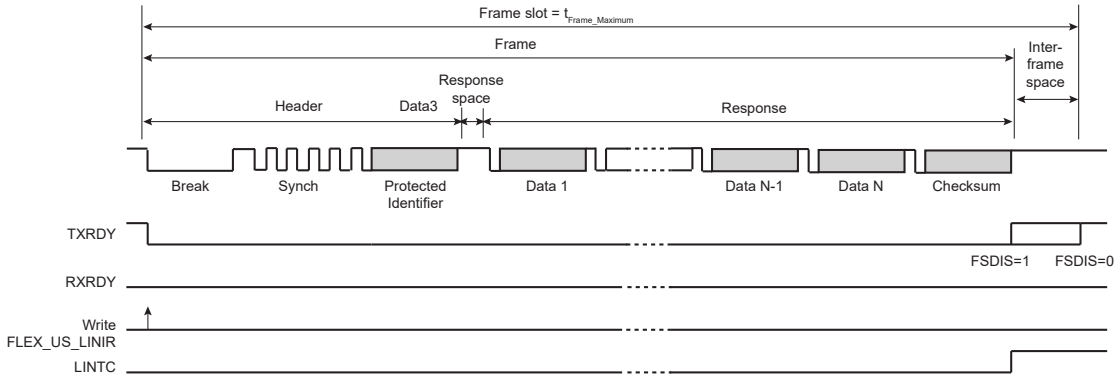



**Figure 46-49. Master Node Configuration, NACT = IGNORE**




#### 46.7.9.15.2 Slave Node Configuration

- Write FLEX_US_CR.TXEN and FLEX_US_CR.RXEN to enable both the transmitter and the receiver.
- Write FLEX_US_MR.USART_MODE to select the LIN mode and the slave node configuration.
- Write FLEX_US_BRGR.CD and FLEX_US_BRGR.FP to configure the baud rate.
- Wait until FLEX_US_CSR.LINID rises.
- Check LINISFE and LINPE errors.
- Read FLEX_US_RHR.IDCHR.
- Write NACT, PARDIS, CHKDIS, CHKTYPE, DLCM and DLC in FLEX_US_LINMR to configure the frame transfer.

IMPORTANT: If the NACT configuration for this frame is PUBLISH, FLEX_US_LINMR must be written with NACT = PUBLISH even if this field is already correctly configured, in order to set the TXREADY flag and the corresponding write transfer request.

What comes next depends on the NACT configuration:

- Case 1: NACT = PUBLISH, the LIN controller sends the response.
  - Wait until FLEX_US_CSR.TXRDY rises.
  - Write FLEX_US_THR.TCHR to send a byte.
  - If all the data have not been written, repeat the two previous steps.
  - Wait until FLEX_US_CSR. LINTC rises.
  - Check the LIN errors.

- Case 2: NACT = SUBSCRIBE, the USART receives the response.
  - Wait until FLEX_US_CSR.RXRDY rises.
  - Read FLEX_US_RHR.RCHR.
  - If all the data have not been read, repeat the two previous steps.
  - Wait until FLEX_US_CSR.LINTC rises.
  - Check the LIN errors.
- Case 3: NACT = IGNORE, the USART is not concerned by the response.
  - Wait until FLEX_US_CSR.LINTC rises.
  - Check the LIN errors.

**Figure 46-50. Slave Node Configuration, NACT = PUBLISH**

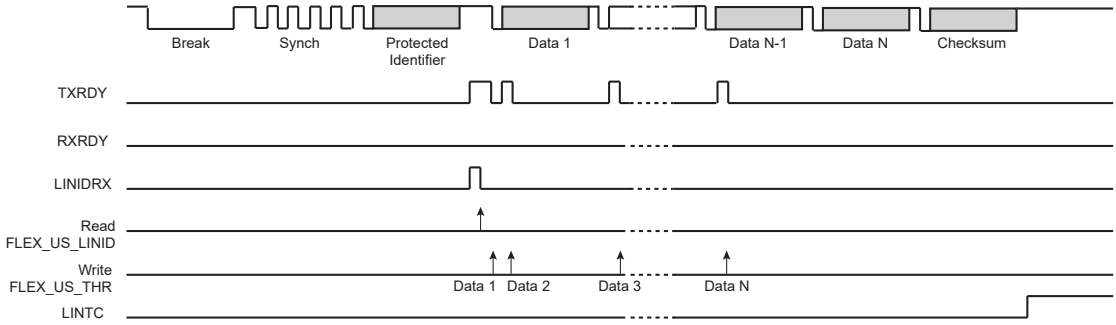



**Figure 46-51. Slave Node Configuration, NACT = SUBSCRIBE**

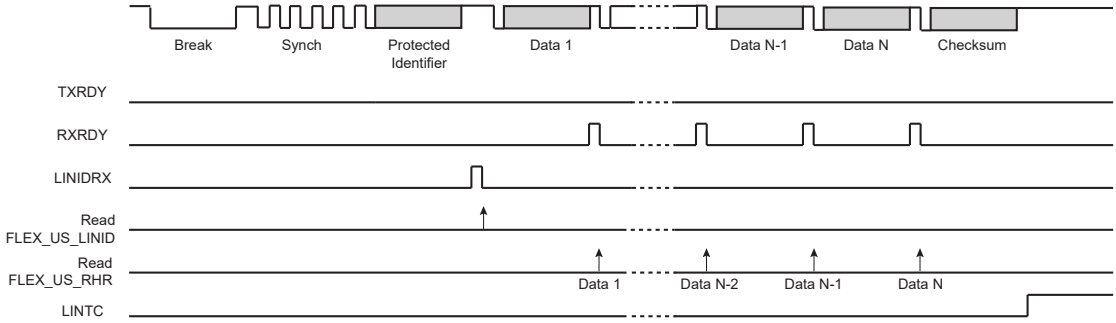
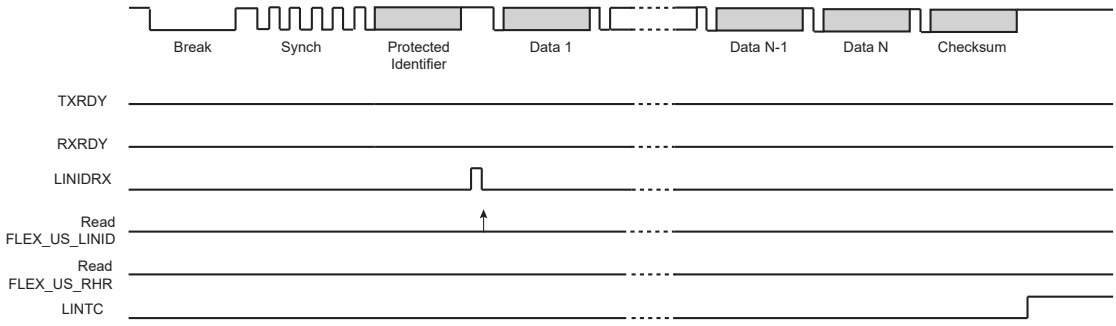



**Figure 46-52. Slave Node Configuration, NACT = IGNORE**




#### 46.7.9.16 LIN Frame Handling with the DMAC

The USART can be used in association with the DMAC in order to transfer data directly into/from the on- and off-chip memories without any processor intervention.

The DMAC uses the trigger flags, TXRDY and RXRDY, to write or read into the USART. The DMAC always writes in the Transmit Holding Register (FLEX_US_THR) and it always reads in the Receive

Holding Register (FLEX_US_RHR). The size of the data written or read by the DMAC in the USART is always a byte.

#### 46.7.9.16.1 Master Node Configuration

The user can choose between two DMAC modes by configuring the FLEX_US_LINMR.PDCM bit:

- PDCM = 1: The LIN configuration is stored in the WRITE buffer and it is written by the DMAC in the Transmit Holding register FLEX_US_THR (instead of the LIN Mode register FLEX_US_LINMR). Because the DMAC transfer size is limited to a byte, the transfer is split into two accesses. During the first access, the NACT, PARDIS, CHKDIS, CHKTYP, DLM and FSDIS bits are written. During the second access, the 8-bit DLC field is written.
- PDCM = 0: The LIN configuration is not stored in the WRITE buffer and it must be written by the user in FLEX_US_LINMR.

The WRITE buffer also contains the Identifier and the data, if the USART sends the response (NACT = PUBLISH).

The READ buffer contains the data if the USART receives the response (NACT = SUBSCRIBE).

**Figure 46-53. Master Node with DMAC (PDCM = 1)**

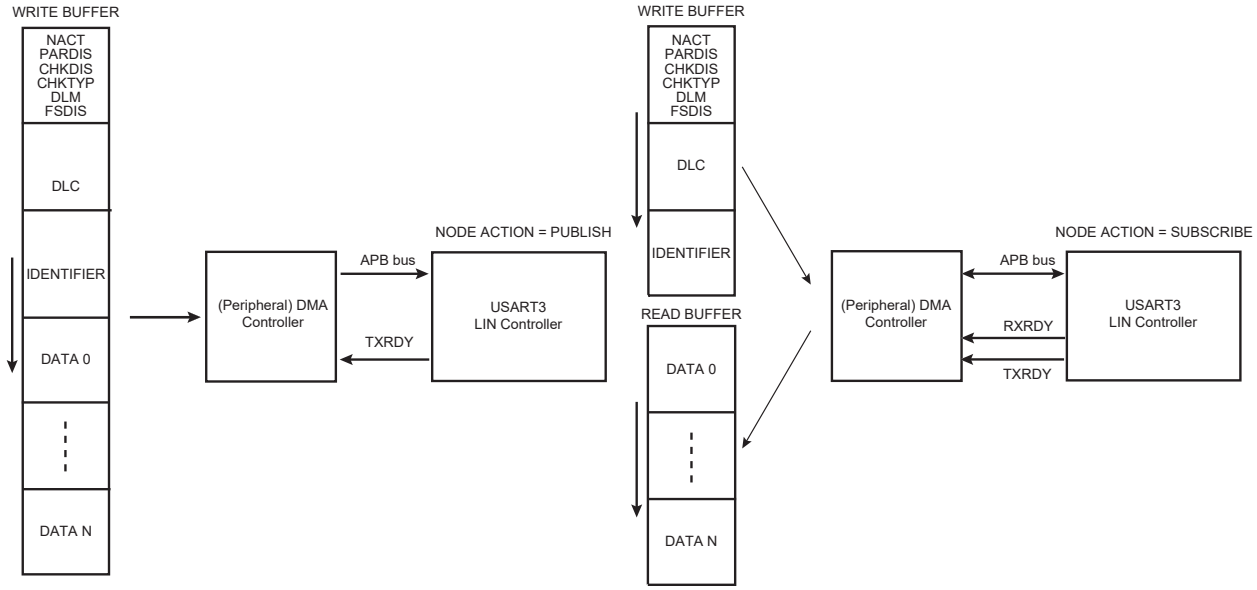
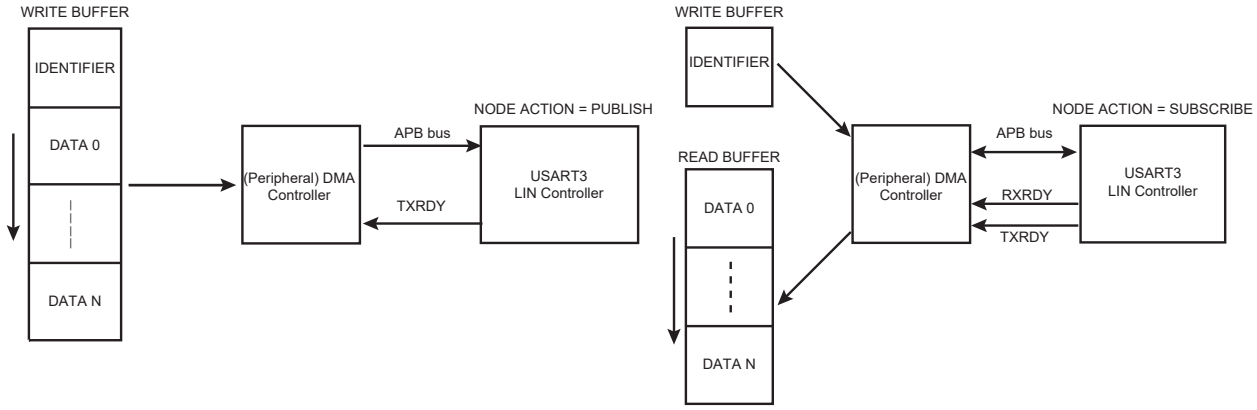



**Figure 46-54. Master Node with DMAC (PDCM = 0)**



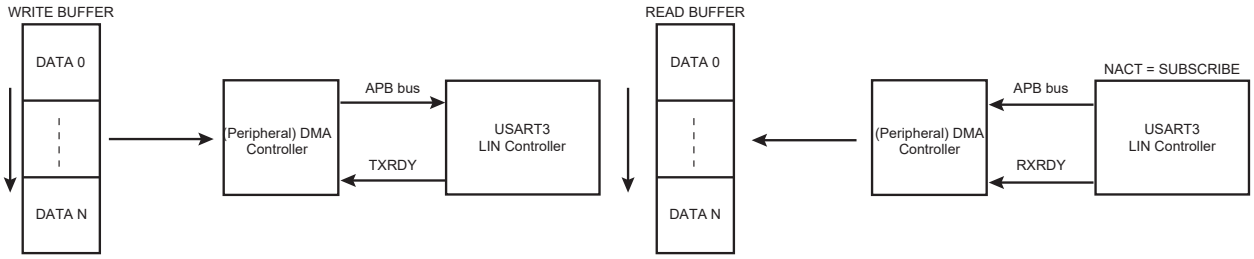
#### 46.7.9.16.2 Slave Node Configuration

In this configuration, the DMAC transfers only the data. The identifier must be read by the user in the LIN Identifier Register (FLEX_US_LINIR). The LIN mode must be written by the user in FLEX_US_LINMR.

The WRITE buffer contains the data if the USART sends the response (NACT = PUBLISH).

The READ buffer contains the data if the USART receives the response (NACT = SUBSCRIBE).

**Figure 46-55. Slave Node with DMAC**




#### 46.7.9.17 Wakeup Request

Any node in a sleeping LIN cluster may request a wakeup.

In the LIN 2.0 specification, the wakeup request is issued by forcing the bus to the dominant state from 250 µs to 5 ms. For this, it is necessary to send the character 0xF0 in order to impose five successive dominant bits. Whatever the baud rate is, this character respects the specified timings.

- Baud rate min = 1 kbit/s -> $t_{bit}$ = 1 ms -> 5 $t_{bit}$ = 5 ms
- Baud rate max = 20 kbit/s -> $t_{bit}$ = 50 µs -> 5 $t_{bit}$ = 250 µs

In the LIN 1.3 specification, the wakeup request should be generated with the character 0x80 in order to impose eight successive dominant bits.

Using the FLEX_US_LINMR.WKUPTYP bit, the user can choose to send either a LIN 2.0 wakeup request (WKUPTYP = 0) or a LIN 1.3 wakeup request (WKUPTYP = 1).

A wakeup request is transmitted by writing the FLEX_US_CR.LINWKUP bit to 1. Once the transfer is completed, the LINTC flag is asserted in the Status Register (FLEX_US_CSR). It is cleared by writing a one to the FLEX_US_CR.RSTSTA bit.

#### 46.7.9.18 Bus Idle Timeout

If the LIN bus is inactive for a certain duration, the slave nodes shall automatically enter in Sleep mode. In the LIN 2.0 specification, this timeout is defined as 4 seconds. In the LIN 1.3 specification, it is defined as 25,000 $t_{bit}$.

In slave Node configuration, the receiver timeout detects an idle condition on the RXD line. When a timeout is detected, the FLEX_US_CSR.TIMEOUT bit rises and can generate an interrupt, thus indicating to the driver to go into Sleep mode.

The timeout delay period (during which the receiver waits for a new character) is programmed in the FLEX_US_RTOR.TO field. If a zero is written to the TO field, the Receiver Timeout is disabled and no timeout is detected. The FLEX_US_CSR.TIMEOUT bit remains at 0. Otherwise, the receiver loads a 17-bit counter with the value programmed in TO. This counter is decremented at each bit period and reloaded each time a new character is received. If the counter reaches 0, the FLEX_US_CSR.TIMEOUT bit rises.

If STTTO is performed, the counter clock is stopped until a first character is received.

If RETTO is performed, the counter starts counting down immediately from the value TO.

**Table 46-14. Receiver Timeout Programming**

| LIN Specification | Baud Rate | Timeout period | TO |
|---|---|---|---|
| 2.0 | 1,000 bit/s | 4s | 4,000 |
| | 2,400 bit/s | | 9,600 |
| | 9,600 bit/s | | 38,400 |
| | 19,200 bit/s | | 76,800 |
| | 20,000 bit/s | | 80,000 |
| 1.3 | – | 25,000 $t_{bit}$ | 25,000 |

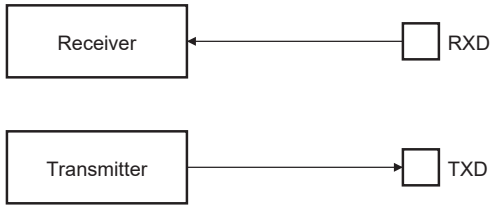
### 46.7.10 Test Modes

The USART can be programmed to operate in three different test modes. The internal loopback capability allows on-board diagnostics. In Loopback mode, the USART interface pins are disconnected or not and reconfigured for loopback internally or externally.

#### 46.7.10.1 Normal Mode

Normal mode connects the RXD pin on the receiver input and the transmitter output on the TXD pin.

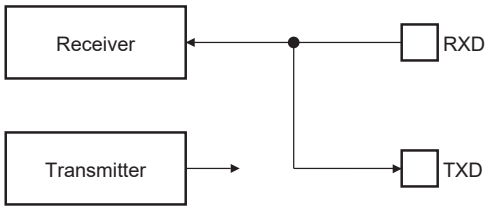
**Figure 46-56. Normal Mode Configuration**




#### 46.7.10.2 Automatic Echo Mode

Automatic Echo mode allows bit-by-bit retransmission. When a bit is received on the RXD pin, it is sent to the TXD pin, as shown in the following figure. Programming the transmitter has no effect on the TXD pin. The RXD pin is still connected to the receiver input, thus the receiver remains active.

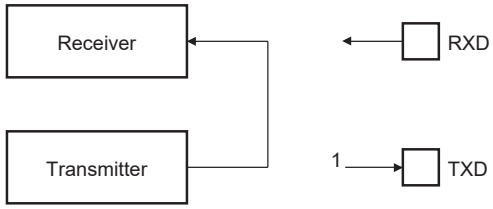
**Figure 46-57. Automatic Echo Mode Configuration**




#### 46.7.10.3 Local Loopback Mode

Local Loopback mode connects the output of the transmitter directly to the input of the receiver, as shown in the following figure. The TXD and RXD pins are not used. The RXD pin has no effect on the receiver and the TXD pin is continuously driven high, as in idle state.

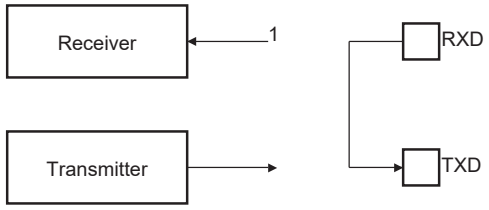
**Figure 46-58. Local Loopback Mode Configuration**




#### 46.7.10.4 Remote Loopback Mode

Remote Loopback mode directly connects the RXD pin to the TXD pin, as shown in the following figure. The transmitter and the receiver are disabled and have no effect. This mode allows bit-by-bit retransmission.

**Figure 46-59. Remote Loopback Mode Configuration**




### 46.7.11 USART FIFOs

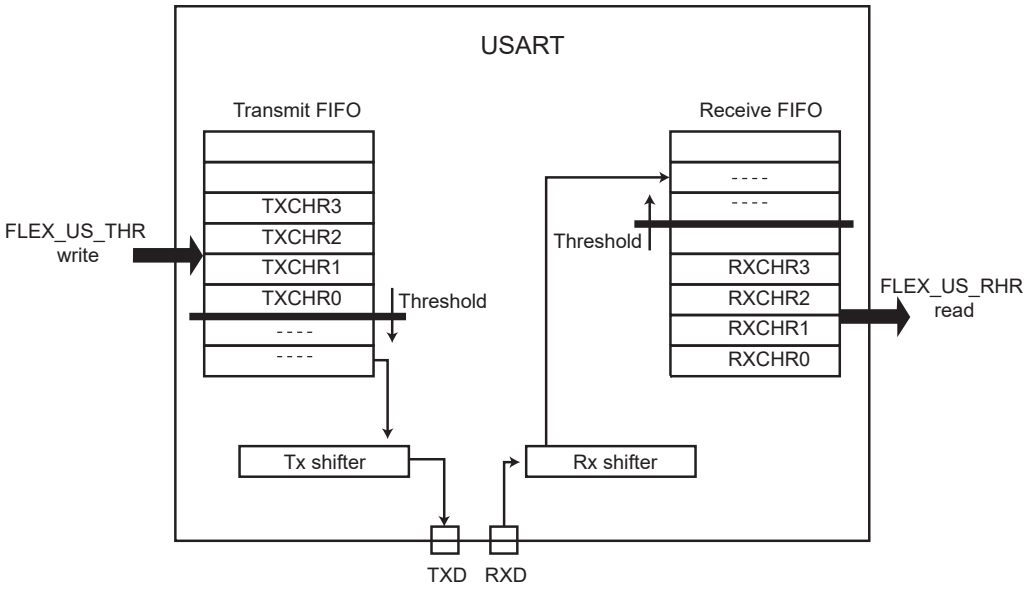
#### 46.7.11.1 Overview

The USART includes two FIFOs which can be enabled/disabled using FLEX_US_CR.FIFOEN/FIFODIS. Both the transmitter and the receiver must be disabled before enabling or disabling the FIFOs, using the FLEX_US_CR.TXDIS/RXDIS bits.

Writing FLEX_US_CR.FIFOEN to ‘1’ enables a 32-data Transmit FIFO and a 32-data Receive FIFO.

It is possible to write or to read single or multiple data in the same access to FLEX_US_THR/RHR. See sections USART Single Data Mode and USART Multiple Data Mode.

**Figure 46-60. FIFOs Block Diagram**



### 46.7.11.2 Sending Data with FIFO Enabled

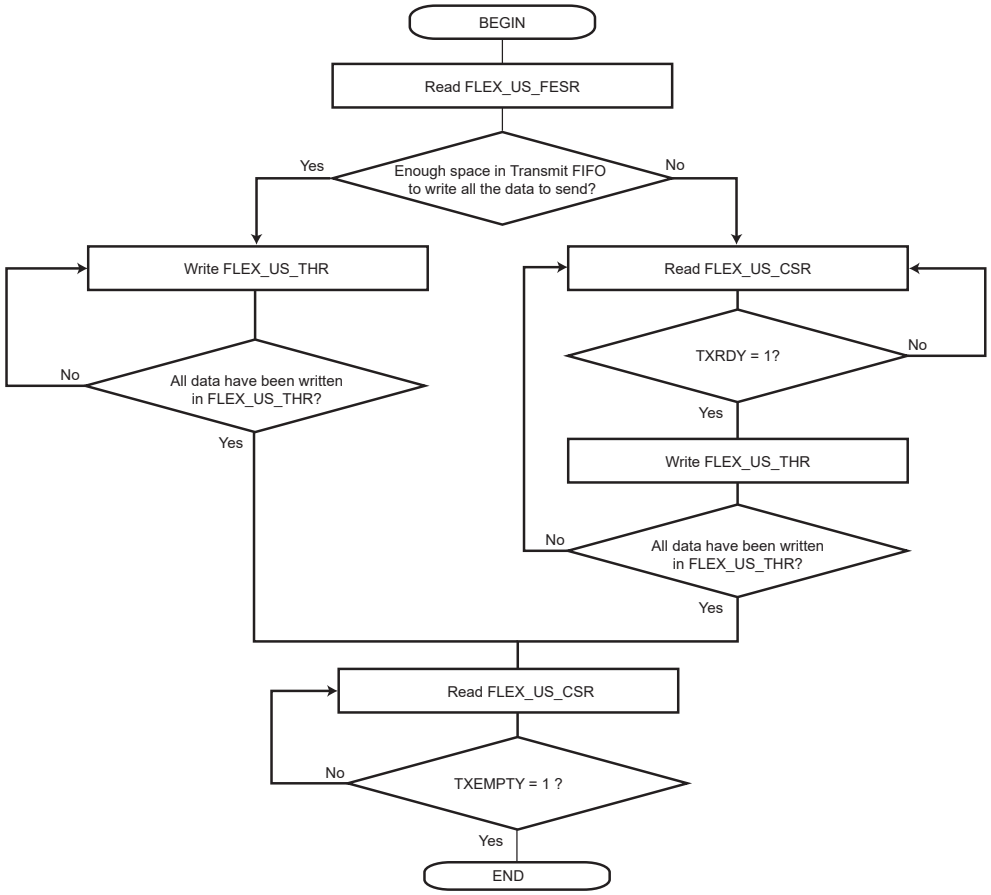
When the Transmit FIFO is enabled, write access to FLEX_US_THR loads the Transmit FIFO.

The FIFO level is provided in FLEX_US_FLR.TXFL. If the FIFO can accept the number of data to be transmitted, there is no need to monitor FLEX_US_CSR.TXRDY and the data can be successively written in FLEX_US_THR.

If the FIFO cannot accept the data due to insufficient space, wait for the TXRDY flag to be set before writing the data in FLEX_US_THR.

When the space in the FIFO allows only a portion of the data to be written, the TXRDY flag must be monitored before writing the remaining data.

**Figure 46-61. Sending Data with FIFO Enabled**

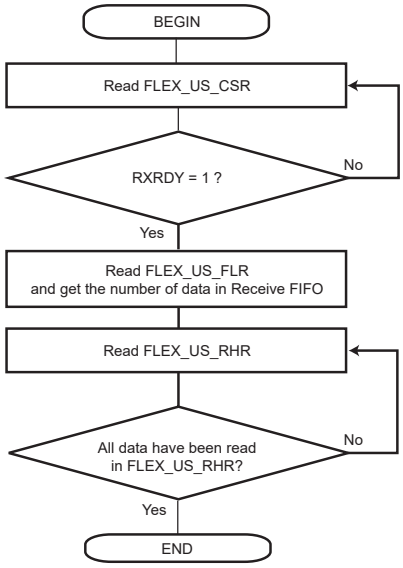



### 46.7.11.3 Receiving Data with FIFO Enabled

When the Receive FIFO is enabled, FLEX_US_RHR access reads the FIFO.

When data are present in the Receive FIFO (RXRDY flag set to ‘1’), the exact number of data can be checked with FLEX_US_FLR.RXFL. All the data can be read successively in FLEX_US_RHR without checking the RXRDY flag between each access.

**Figure 46-62. Receiving Data with FIFO Enabled**




#### 46.7.11.4 Clearing/Flushing FIFOs

Each FIFO can be cleared/flushed using FLEX_US_CR.TXFCLR/RXFCLR.

#### 46.7.11.5 TXEMPTY, TXRDY and RXRDY Behavior

FLEX_US_CSR.TXEMPTY, FLEX_US_CSR.TXRDY and FLEX_US_CSR.RXRDY flags display a specific behavior when FIFOs are enabled.

The TXEMPTY flag is cleared as long as there are characters in the Transmit FIFO or in the internal shift register. TXEMPTY is set when there are no characters in the Transmit FIFO and in the internal shift register.

TXRDY indicates if a data can be written in the Transmit FIFO. Thus the TXRDY flag is set as long as the Transmit FIFO can accept new data. See figure TXRDY in Single Data Mode and TXRDYM = 0.

RXRDY indicates if an unread data is present in the Receive FIFO. Thus the RXRDY flag is set as soon as one unread data is in the Receive FIFO. See figure RXRDY in Single Data Mode and RXRDYM = 0 below.

TXRDY and RXRDY behavior can be modified using the TXRDYM and RXRDYM fields in the USART FIFO Mode Register (FLEX_US_FMR) to reduce the number of accesses to FLEX_US_RHR/THR. However, for some configurations, the following constraints apply:

- If FLEX_US_MR.MODE9 is set, FLEX_US_FMR.TXRDYM/RXRDYM must be cleared.
- If FLEX_US_MR.USART_MODE is set to either LIN_MASTER or LIN_SLAVE, FLEX_US_FMR.TXRDYM/RXRDYM must be cleared.

See USART FIFO Mode Register for the FIFO configuration.

**Figure 46-63. TXRDY in Single Data Mode and TXRDYM = 0**

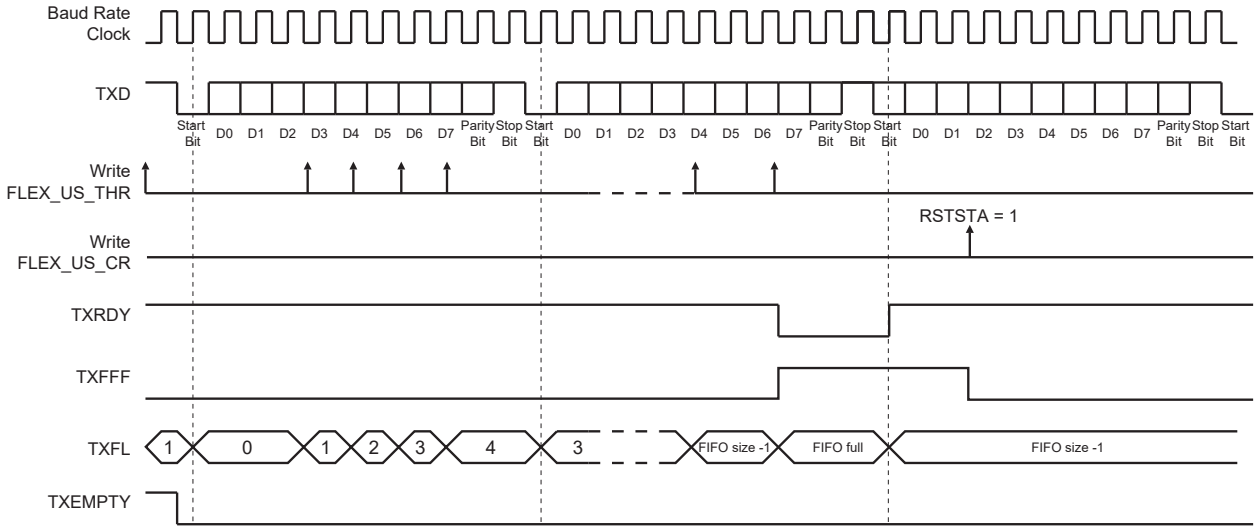
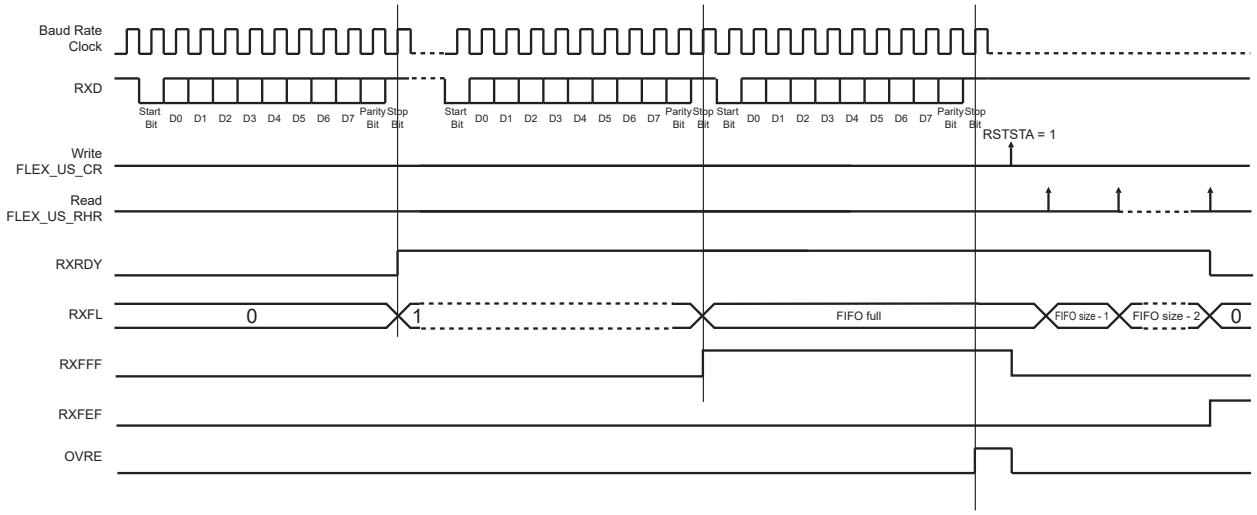



**Figure 46-64. RXRDY in Single Data Mode and RXRDYM = 0**




### 46.7.11.6 USART Single Data Mode

In Single Data mode, only one data is written every time FLEX_US_THR is accessed, and only one data is read every time FLEX_US_RHR is accessed.

When FLEX_US_FMR.TXRDYM = 0, the Transmit FIFO operates in Single Data mode.

When FLEX_US_FMR.RXRDYM = 0, the Receive FIFO operates in Single Data mode.

If FLEX_US_MR.MODE9 is set, or if FLEX_US_MR.USART_MODE is set to either LIN_MASTER or LIN_SLAVE, the FIFOs must operate in Single Data mode.

See USART Receive Holding Register (FLEX_US_RHR) and USART Transmit Holding Register (FLEX_US_THR).

#### 46.7.11.6.1 DMAC

The DMAC transfer type must be configured in bytes or halfwords when FIFOs operate in Single Data mode (the same applies when FIFOs are disabled).

#### 46.7.11.7 USART Multiple Data Mode

Multiple Data mode minimizes the number of accesses by concatenating the data to send/read in one access.

When FLEX_US_FMR.TXRDYM > 0, the Transmit FIFO operates in Multiple Data mode.

When FLEX_US_FMR.RXRDYM > 0, the Receive FIFO operates in Multiple Data mode.

However, Multiple Data mode cannot be used for the following configurations:

- If FLEX_US_MR.MODE9 is set
- If FLEX_US_MR.USART_MODE is set to either LIN_MASTER or LIN_SLAVE

In Multiple Data mode, it is possible to write/read up to four data in one FLEX_US_THR/FLEX_US_RHR access.

The number of data to write/read is defined by the size of the register access. If the access is a byte-size register access, only one data is written/read, if the access is a halfword size register access, then two data are written/read and, finally, if the access is a word-size register access, four data are written/read.

Written/read data are always right-aligned, as described in sections USART Receive Holding Register (FIFO Multi Data) and USART Transmit Holding Register (FIFO Multi Data).

As an example, if the Transmit FIFO is empty and there are six data to send, either of the following write accesses may be performed:

- six FLEX_US_THR-byte write accesses
- three FLEX_US_THR-halfword write accesses
- one FLEX_US_THR word write access and one FLEX_US_THR halfword write access

With a Receive FIFO containing six data, any of the following read accesses may be performed:

- six FLEX_US_RHR-byte read accesses
- three FLEX_US_RHR-halfword read accesses
- one FLEX_US_RHR-word read access and one FLEX_US_RHR-halfword read access

##### 46.7.11.7.1 TXRDY and RXRDY Configuration

In Multiple Data mode, it is possible to write one or more data in the same FLEX_US_THR/FLEX_US_RHR access. The TXRDY flag indicates if one or more data can be written in the FIFO depending on the configuration of FLEX_US_FMR.TXRDYM/RXRDYM.

As an example, if four data are written each time in FLEX_US_THR, it is useful to configure the TXRDYM field to the value ‘2’ so that the TXRDY flag is at ‘1’ only when at least four data can be written in the Transmit FIFO.

In the same way, if four data are read each time in FLEX_US_RHR, it is useful to configure the RXRDYM field to the value ‘2’ so that the RXRDY flag is at ‘1’ only when at least four unread data are in the Receive FIFO.

##### 46.7.11.7.2 DMAC

When FIFOs operate in Multiple Data mode, the DMAC transfer type must be configured in byte, halfword or word depending on the FLEX_US_FMR.TXRDYM/RXRDYM settings.

#### 46.7.11.8 Transmit FIFO Lock

- LIN Mode:

If a frame is aborted using the Abort LIN Transmission bit (FLEX_US_CR.LINABT), a lock is set on the Transmit FIFO, preventing any new frame from being sent until it is cleared. This allows clearing the FIFO if needed, resetting DMAC channels, etc., without any risk.

The TXFLOCK bit in the USART FIFO Event Status Register (FLEX_US_FESR) is used to check the state of the Transmit FIFO lock.

The Transmit FIFO lock can be cleared by setting FLEX_US_CR.TXFLCLR to ‘1’.

#### 46.7.11.9 FIFO Pointer Error

A FIFO overflow is reported in FLEX_US_FESR.

If the Transmit FIFO is full and a write access is performed on FLEX_US_THR, it generates a Transmit FIFO pointer error and sets FLEX_US_FESR.TXFPTEF.

In Multiple Data mode, if the number of data written in FLEX_US_THR (according to the register access size) is greater than the free space in the Transmit FIFO, a Transmit FIFO pointer error is generated and FLEX_US_FESR.TXFPTEF is set.

A FIFO underflow is reported in FLEX_US_FESR.

In Multiple Data mode, if the number of data read in FLEX_US_RHR (according to the register access size) is greater than the number of unread data in the Receive FIFO, a Receive FIFO pointer error is generated and FLEX_US_FESR.RXFPTEF is set.

No pointer error occurs if the FIFO state/level is checked before writing/reading in FLEX_US_THR/ FLEX_US_RHR. The FIFO state/level can be checked either with TXRDY, RXRDY, TXFL or RXFL. When a pointer error occurs, other FIFO flags may not behave as expected; their states should be ignored.

If a Transmit pointer error occurs, a transmitter reset must be performed using FLEX_US_CR.RSTTX. If a Receive pointer error occurs, a receiver reset must be performed using FLEX_US_CR.RSTRX.

#### 46.7.11.10 FIFO Thresholds

Each Transmit and Receive FIFO includes a threshold feature used to set a flag and an interrupt when a FIFO threshold is crossed. Thresholds are defined as a number of data in the FIFO, and the FIFO state (TXFL or RXFL) represents the number of data currently in the FIFO.

The Transmit FIFO threshold can be set using the field FLEX_US_FMR.TXFTHRES. Each time the Transmit FIFO level goes from ‘above threshold’ to ‘equal to or below threshold’, the flag FLEX_US_FESR.TXFTHF is set. The application is warned that the Transmit FIFO has reached the defined threshold and that it can be reloaded.

The Receive FIFO threshold can be set using the field FLEX_US_FMR.RXFTHRES. Each time the Receive FIFO level goes from ‘below threshold’ to ‘equal to or above threshold’, the flag FLEX_US_FESR.RXFTHF is set. The application is warned that the Receive FIFO has reached the defined threshold and that it can be read to prevent an underflow.

The Receive FIFO threshold 2 can be set using the field FLEX_US_FMR.RXFTHRES2. Each time the Receive FIFO level goes from ‘above threshold 2’ to ‘equal to or below threshold 2’, the flag FLEX_US_FESR.RXFTHF2 is set. The application is warned that the Receive FIFO has reached the defined threshold and that it can be read to prevent an underflow.

The TXFTHF, RXFTHF and RXTHF2 flags can be configured to generate an interrupt using FLEX_US_FIER and FLEX_US_FIDR.

#### 46.7.11.11 FIFO Flags

FIFOs come with a set of flags which can be configured to generate interrupts through FLEX_US_FIER and FLEX_US_FIDR.

FIFO flags state can be read in FLEX_US_FESR. They are cleared by writing FLEX_US_CR.RSTSTA to ‘1’.

### 46.7.12 USART Register Write Protection

The FLEXCOM operating mode (FLEX_MR.OPMODE) must be set to FLEX_MR_OPMODE_USART to enable access to the write protection registers.

To prevent any single software error from corrupting USART behavior, certain registers in the address space can be write-protected by setting the WPEN (Write Protection Enable) bit in the USART Write Protection Mode Register (FLEX_US_WPMR).

If a write access to a write-protected register is detected, the Write Protection Violation Status (WPVS) flag in the USART Write Protection Status Register (FLEX_US_WPSR) is set and the Write Protection Violation Source (WPVSRC) field indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading FLEX_US_WPSR.

The following registers can be write-protected when WPEN is set:

- USART Mode Register
- USART Baud Rate Generator Register
- USART Receiver Timeout Register
- USART Transmitter Timeguard Register
- USART FI DI RATIO Register
- USART IrDA FILTER Register
- USART Manchester Configuration Register
- USART Comparison Register

## 46.8 SPI Functional Description

### 46.8.1 Modes of Operation

The SPI operates in Master mode or in Slave mode.

- The SPI operates in Master mode by writing a 1 to the MSTR bit in the SPI Mode Register (FLEX_SPI_MR):
  - The pins NPCS0 to NPCS1 are all configured as outputs.
  - The SPCK pin is driven.
  - The MISO line is wired on the receiver input.
  - The MOSI line is driven as an output by the transmitter.
- The SPI operates in Slave mode if the MSTR bit in FLEX_SPI_MR is written to 0:
  - The MISO line is driven by the transmitter output.
  - The MOSI line is wired on the receiver input.
  - The SPCK pin is driven by the transmitter to synchronize the receiver.
  - The NPCS0 pin becomes an input, and is used as a slave select signal (NSS).
  - Pin NPCS1 is not driven and can be used for other purposes.

The data transfers are identically programmable for both modes of operation. The bit rate generator is activated only in Master mode.

### 46.8.2 Data Transfer

Four combinations of polarity and phase are available for data transfers. The clock polarity is programmed with the CPOL bit in the SPI Chip Select Register (FLEX_SPI_CSR). The clock phase is programmed with the NCPHA bit. These two parameters determine the edges of the clock signal on which data are driven and sampled. Each of the two parameters has two possible states, resulting in four possible combinations that are incompatible with one another. Consequently, a master/slave pair must use the same parameter pair values to communicate. If multiple slaves are connected and require different configurations, the master must reconfigure itself each time it needs to communicate with a different slave.

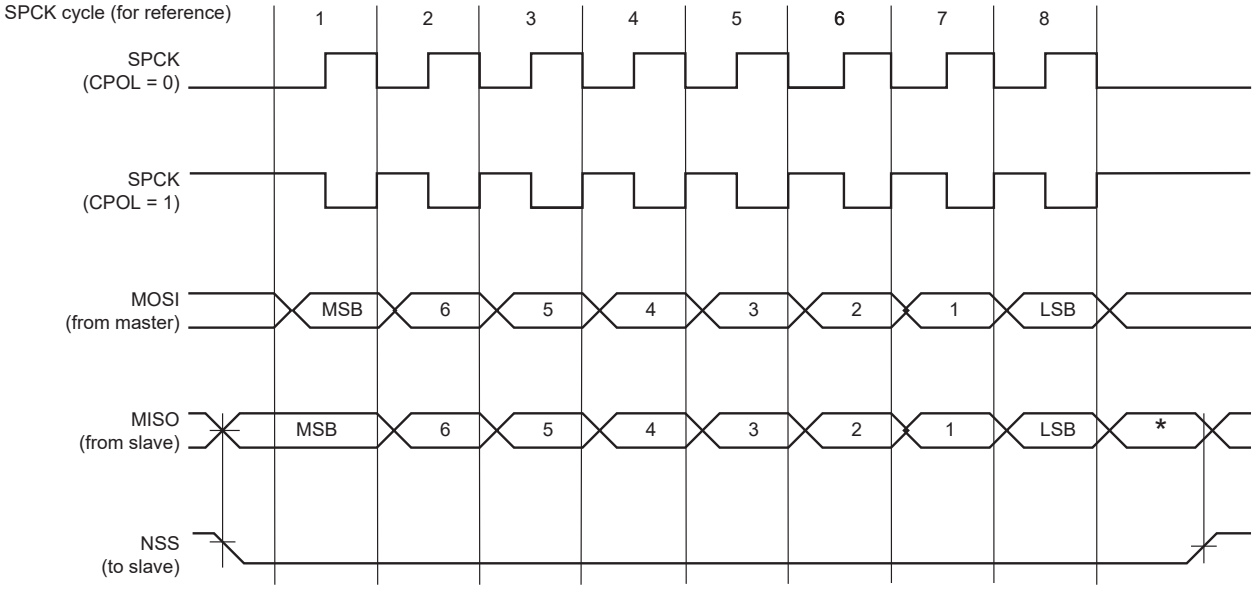
The following table shows the four modes and corresponding parameter settings.

**Table 46-15. SPI Bus Protocol Mode**

| SPI Mode | CPOL | NCPHA | Shift SPCK Edge | Capture SPCK Edge | SPCK Inactive Level |
|---|---|---|---|---|---|
| 0 | 0 | 1 | Falling | Rising | Low |
| 1 | 0 | 0 | Rising | Falling | Low |
| 2 | 1 | 1 | Rising | Falling | High |
| 3 | 1 | 0 | Falling | Rising | High |

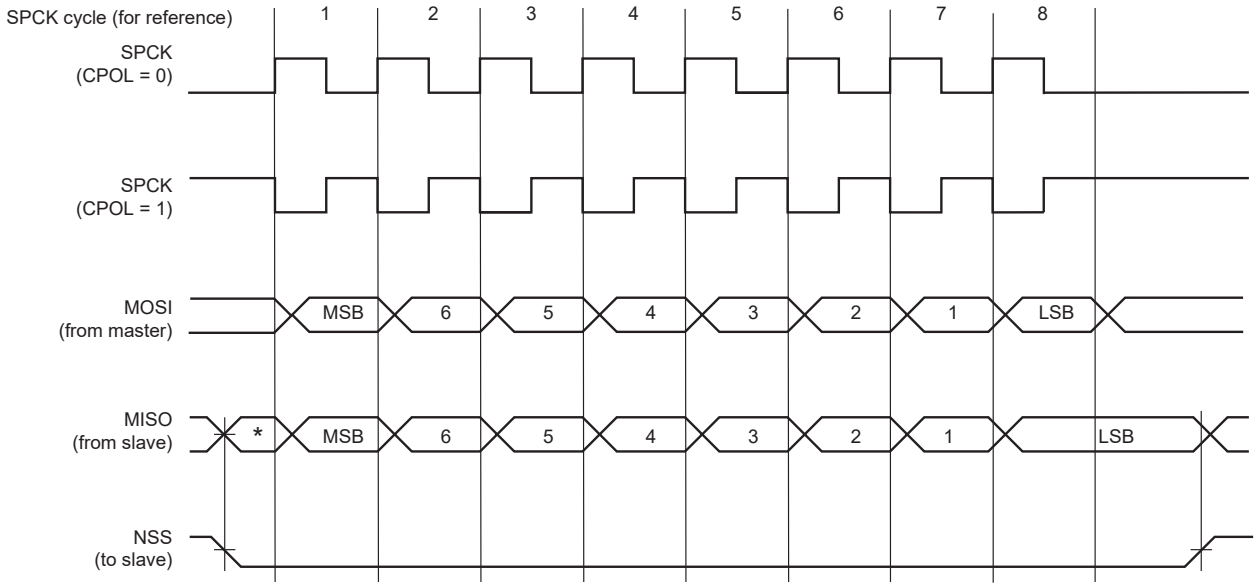
The following figures show examples of data transfers.

**Figure 46-65. SPI Transfer Format (NCPHA = 1, 8 bits per transfer)**




* Not defined.

**Figure 46-66. SPI Transfer Format (NCPHA = 0, 8 bits per transfer)**




* Not defined.

### 46.8.3 Master Mode Operations

When configured in Master mode, the SPI operates on the clock generated by the internal programmable bit rate generator. It fully controls the data transfers to and from the slave(s) connected to the SPI bus. The SPI drives the chip select line to the slave and the serial clock signal (SPCK).

The SPI features two holding registers, the Transmit Data Register (FLEX_SPI_TDR) and the Receive Data Register (FLEX_SPI_RDR), and a single shift register. The holding registers maintain the data flow at a constant rate.

After enabling the SPI, a data transfer starts when the processor writes to FLEX_SPI_TDR. The written data are immediately transferred in the shift register and the transfer on the SPI bus starts. While the data in the shift register is shifted on the MOSI line, the MISO line is sampled and shifted in the shift register. Data cannot be loaded in FLEX_SPI_RDR without transmitting data. If there is no data to transmit, a dummy data can be used (FLEX_SPI_TDR filled with ones). When the WDRBT bit is set, a new data cannot be transmitted if FLEX_SPI_RDR has not been read. If Receiving mode is not required, for example when communicating with a slave receiver only (such as an LCD), the receive status flags in the SPI Status Register (FLEX_SPI_SR) can be discarded.

Before writing the TDR, the FLEX_SPI_MR.PCS field must be set in order to select a slave.

If new data are written in FLEX_SPI_TDR during the transfer, it is kept in FLEX_SPI_TDR until the current transfer is completed. Then, the received data are transferred from the shift register to FLEX_SPI_RDR, the data in FLEX_SPI_TDR is loaded in the shift register and a new transfer starts.

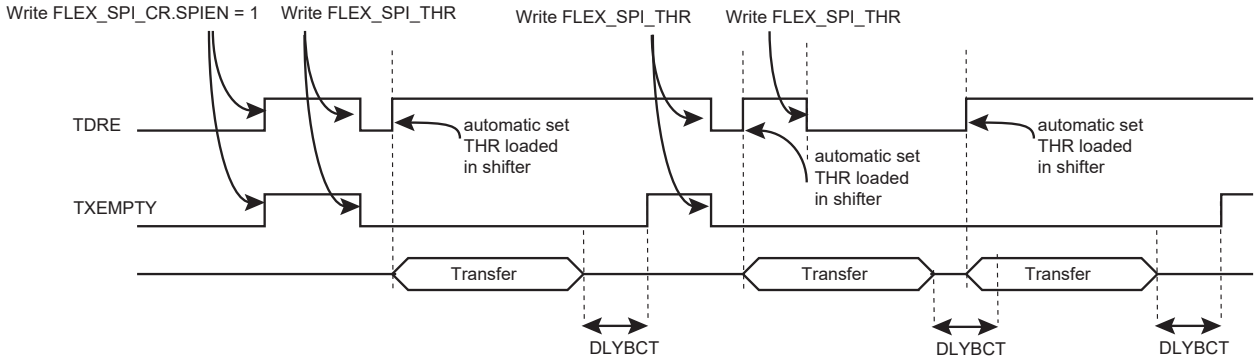
As soon as the FLEX_SPI_TDR is written, the Transmit Data Register Empty (TDRE) flag in FLEX_SPI_SR is cleared. When the data written in FLEX_SPI_TDR is loaded into the shift register, the FLEX_SPI_SR.TDRE flag is set. The TDRE bit is used to trigger the Transmit DMA channel (see figure below).

The end of transfer is indicated by FLEX_SPI_SR.TXEMPTY. If a transfer delay (DLYBCT) is greater than 0 for the last transfer, TXEMPTY is set after the completion of this delay. The peripheral clock can be switched off at this time.

**Note:**

1. When the SPI is enabled, the TDRE and TXEMPTY flags are set.
2. The TXEMPTY flag alone cannot be used to detect the end of the buffer DMA transfer.

**Figure 46-67. TDRE and TXEMPTY Flag Behavior**



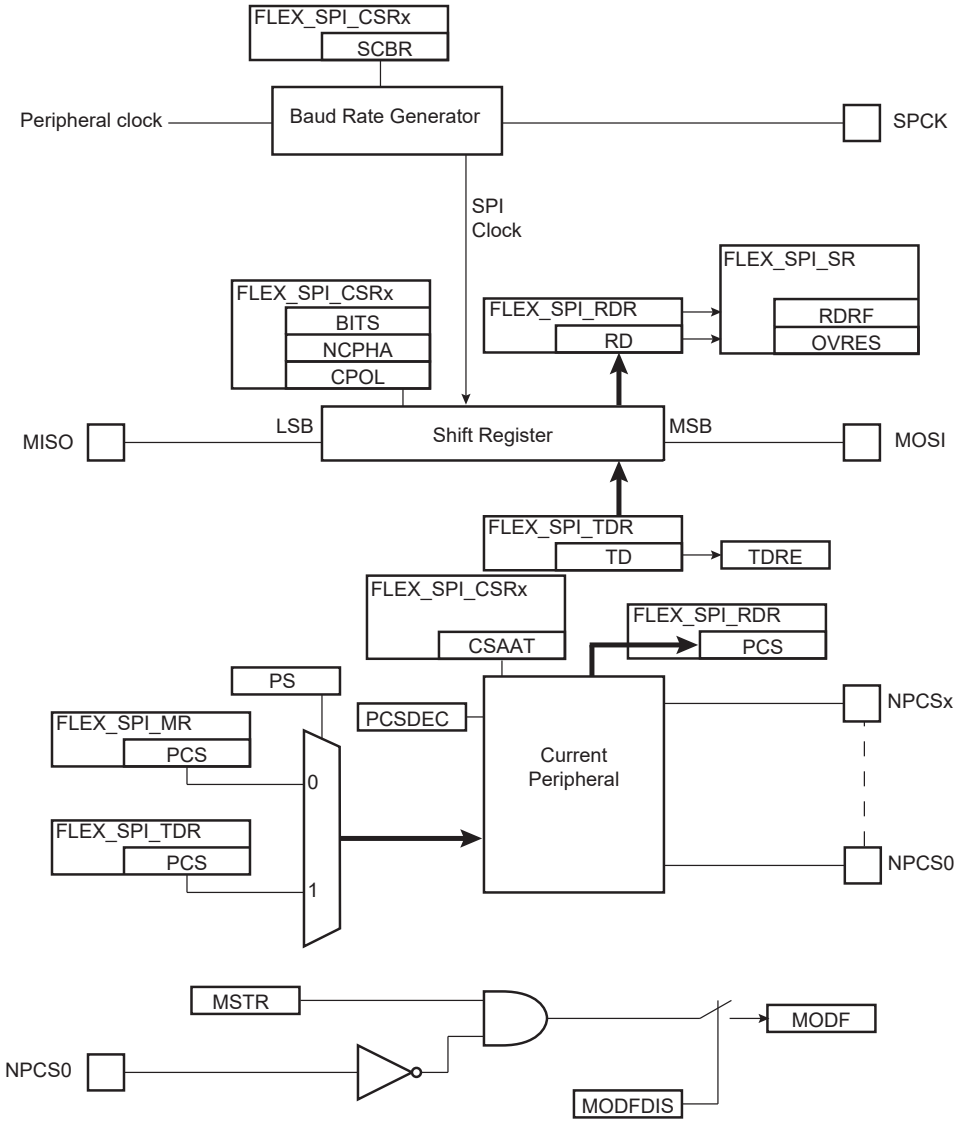

The transfer of received data from the shift register to FLEX_SPI_RDR is indicated by the Receive Data Register Full (RDRF) bit in FLEX_SPI_SR. When the received data are read, the RDRF bit is cleared.

If FLEX_SPI_RDR has not been read before new data are received, the Overrun Error bit (OVRES) in FLEX_SPI_SR is set. As long as this flag is set, data are loaded in FLEX_SPI_RDR. The user has to read the status register to clear the OVRES bit.

The following figures show, respectively, a block diagram of the SPI when operating in Master mode and a flow chart describing how transfers are handled.

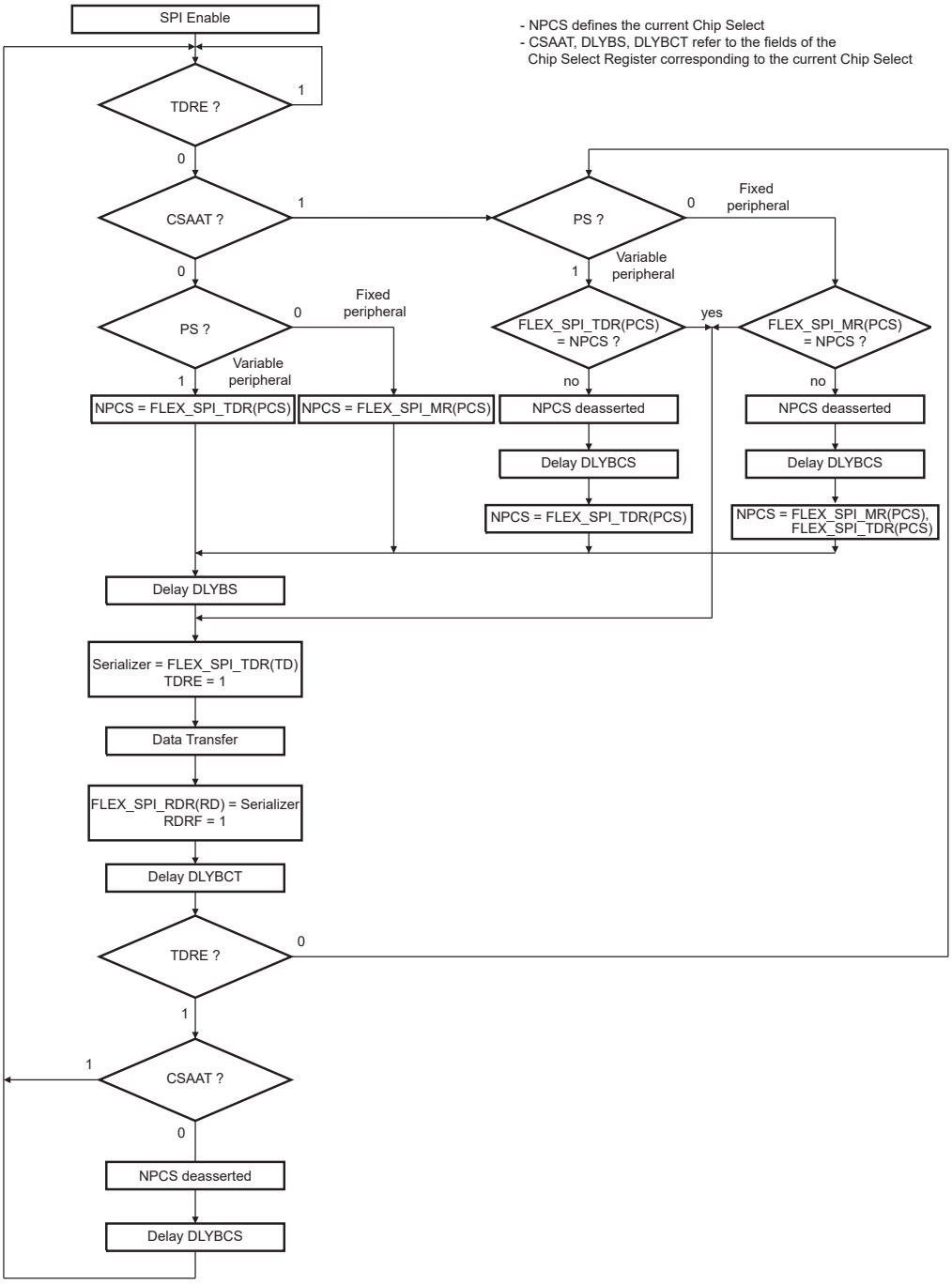
#### 46.8.3.1 Master Mode Block Diagram

**Figure 46-68. Master Mode Block Diagram**



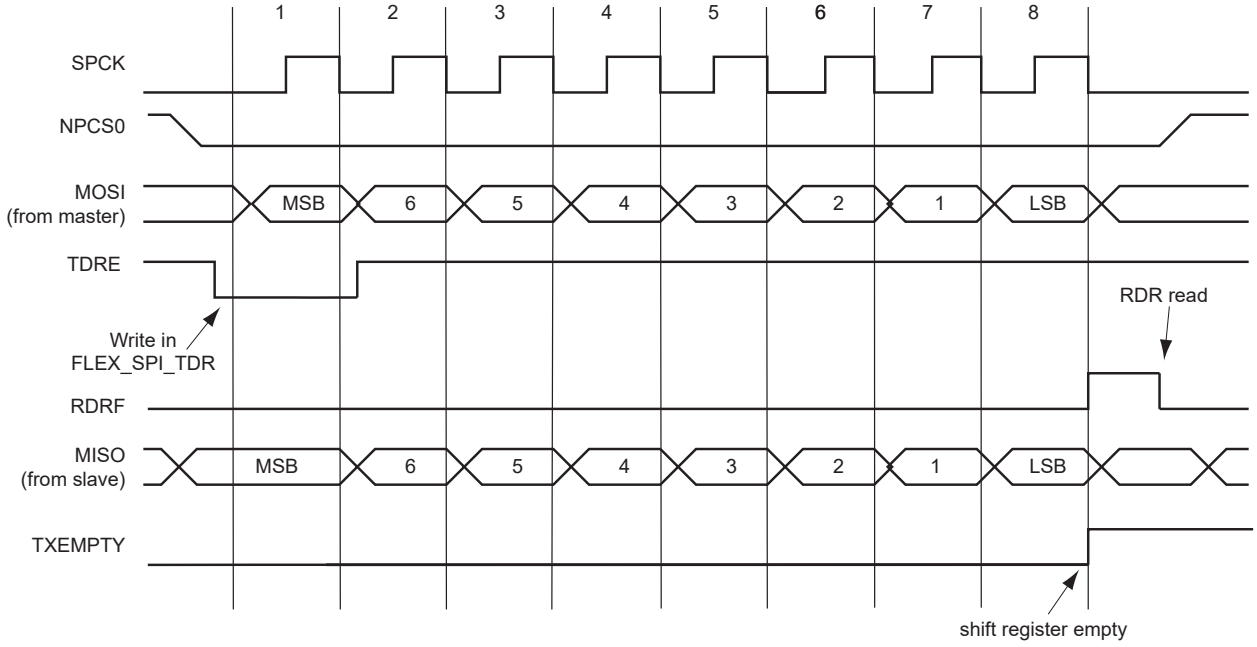
### 46.8.3.2 Master Mode Flowchart

**Figure 46-69. Master Mode**




The following figure shows the behavior of Transmit Data Register Empty (TDRE), Receive Data Register (RDRF) and Transmission Register Empty (TXEMPTY) status flags within FLEX_SPI_SR during an 8-bit data transfer in Fixed mode without the DMAC involved.

**Figure 46-70. Status Register Flags Behavior**




#### 46.8.3.3 Clock Generation

The SPI bit rate clock is generated by dividing a source clock which can be the peripheral clock or a programmable clock from the GCLK. The divider can be a value between 1 and 255.

If the SCBR field is programmed to 1 and the clock source is GCLK, the operating bit rate is peripheral clock (refer to the section “Electrical Characteristics” for the SPCK maximum frequency). Triggering a transfer while SCBR is at 0 can lead to unpredictable results.

At reset, SCBR is 0 and the user has to program it to a valid value before performing the first transfer.

The divisor can be defined independently for each chip select, as it has to be programmed in the FLEX_SPI_CSR.SCBR field. This allows the SPI to automatically adapt the bit rate for each interfaced peripheral without reprogramming.

If GCLK is selected as source clock (FLEX_SPI_MR.BRSRCCLK = 1), the bit rate is independent of the processor/bus clock. Thus, the processor clock can be changed while SPI is enabled. The processor clock frequency changes must be performed only by programming the PMC_MCKR.PRES field (refer to the section “Power Management Controller” (PMC)). Any other method to modify the processor/bus clock frequency (PLL multiplier, etc.) is forbidden when SPI is enabled.

The peripheral clock frequency must be at least three times higher than GCLK.

#### 46.8.3.4 Transfer Delays

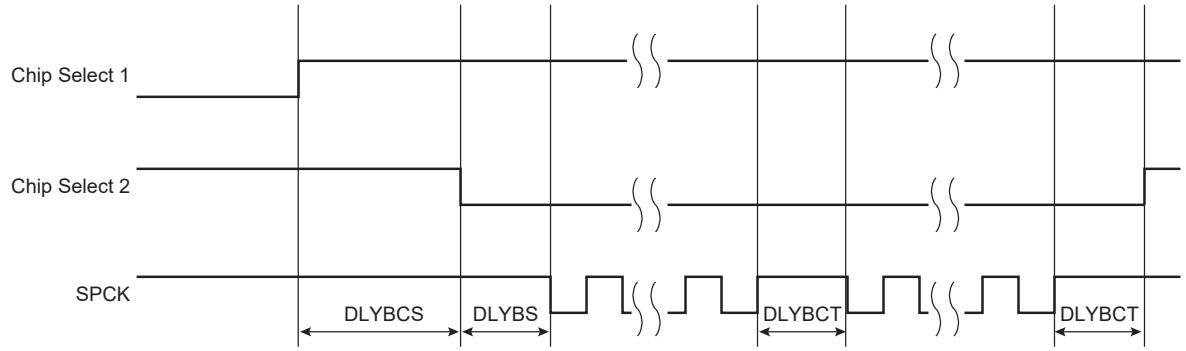
The figure below shows a chip select transfer change and consecutive transfers on the same chip select. Three delays can be programmed to modify the transfer waveforms:

- The delay between the chip selects. It is programmable only once for all chip selects by writing the FLEX_SPI_MR.DLYBCS field. The SPI slave device deactivation delay is managed through DLYBCS. If there is only one SPI slave device connected to the master, the DLYBCS field does not need to be configured. If several slave devices are connected to a master, DLYBCS must be configured depending on the highest deactivation delay. Refer to the SPI slave device electrical characteristics.

- The delay before SPCK, independently programmable for each chip select by writing the DLYBS field. The SPI slave device activation delay is managed through DLYBS. Refer to the SPI slave device electrical characteristics to define DLYBS.
- The delay between consecutive transfers, independently programmable for each chip select by writing the DLYBCT field. The time required by the SPI slave device to process received data is managed through DLYBCT. This time depends on the SPI slave system activity.

These delays allow the SPI to be adapted to the interfaced peripherals and their speed and bus release time.

**Figure 46-71. Programmable Delays**




#### 46.8.3.5 Peripheral Selection

The serial peripherals are selected through the assertion of the NPCS0 to NPCS1 signals. By default, all NPCS signals are high before and after each transfer.

- Fixed Peripheral Select Mode: SPI exchanges data with only one peripheral.
  Fixed Peripheral Select mode is enabled by writing the FLEX_SPI_MR.PS bit to zero. In this case, the current peripheral is defined by the FLEX_SPI_MR.PCS field, and the FLEX_SPI_TDR. PCS field has no effect.
- Variable Peripheral Select Mode: Data can be exchanged with more than one peripheral without having to reprogram FLEX_SPI_MR.PCS.
  Variable Peripheral Select Mode is enabled by setting the FLEX_SPI_MR.PS bit to one. The FLEX_SPI_TDR.PCS field is used to select the current peripheral. This means that the peripheral selection can be defined for each new data. The value must be written in a single access to FLEX_SPI_TDR in the following format:

  [xxxxxxx(7-bit) + LASTXFER(1-bit)(1)+ xxxx(4-bit) + PCS (4-bit) + TD (8 to 16-bit data)]
  with LASTXFER at 0 or 1 depending on the CSAAT bit, and PCS equal to the chip select to assert, as defined in section SPI Transmit Data Register (FLEX_SPI_TDR).
  Note: 1. Optional

  The CSAAT, LASTXFER and CSNAAT bits are discussed in section Peripheral Deselection with DMA.

  If LASTXFER is used, the command must be issued after writing the last character. Instead of LASTXFER, the user can use the SPIDIS command. After the end of the DMA transfer, it is necessary to wait for the TXEMPTY flag and then write SPIDIS into the SPI Control Register (FLEX_SPI_CR). This does not change the configuration register values). The NPCS is disabled after the last character transfer. Then, another DMA transfer can be started if the FLEX_SPI_CR.SPIEN bit has previously been written.

#### 46.8.3.6 SPI Direct Access Memory Controller (DMAC)

In both Fixed and Variable modes, the Direct Memory Access Controller (DMAC) can be used to reduce processor overhead.

The fixed peripheral selection allows buffer transfers with a single peripheral. Using the DMAC is an optimal means, as the size of the data transfer between the memory and the SPI is either 8 bits or 16 bits. However, if the peripheral selection is modified, FLEX_SPI_MR must be reprogrammed.

The variable peripheral selection allows buffer transfers with multiple peripherals without reprogramming FLEX_SPI_MR. Data written in FLEX_SPI_TDR is 32 bits wide and defines the real data to be transmitted and the destination peripheral. Using the DMAC in this mode requires 32-bit wide buffers, with the data in the LSBs and the PCS and LASTXFER fields in the MSBs. However, the SPI still controls the number of bits (8 to 16) to be transferred through MISO and MOSI lines with the chip select configuration registers. This is not the optimal means in terms of memory size for the buffers, but it provides a very effective means to exchange data with several peripherals without any intervention of the processor.

#### 46.8.3.7 Peripheral Chip Select Decoding

The user can program the SPI to operate with up to 3 slave peripherals by decoding the two chip select lines, NPCS0 to NPCS1 with an external decoder/demultiplexer (see the following figure). This can be enabled by setting the FLEX_SPI_MR.PCSDEC bit.

When operating without decoding, the SPI makes sure that in any case only one chip select line is activated, i.e., one NPCS line driven low at a time. If two bits are defined low in a PCS field, only the lowest numbered chip select is driven low.

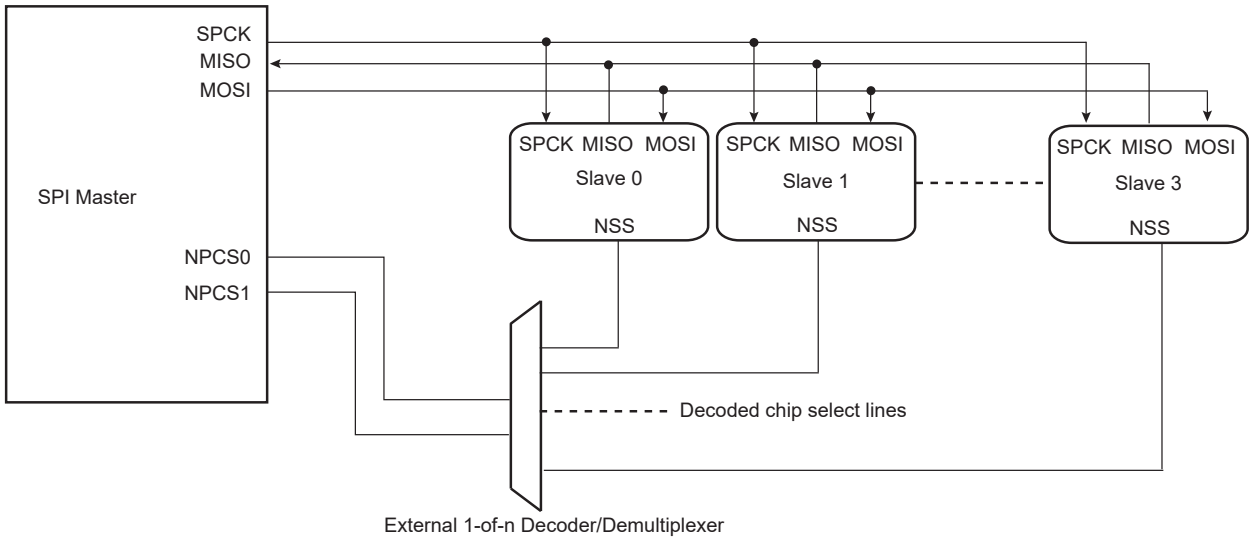
When operating with decoding, the SPI directly outputs the value defined by the PCS field on the NPCS lines of either FLEX_SPI_MR or FLEX_SPI_TDR (depending on PS).

As the SPI sets a default value of 0x3 on the chip select lines (i.e., all chip select lines at 1) when not processing any transfer, only 3 peripherals can be decoded.

The SPI has only two Chip Select registers. As a result, when external decoding is activated, each NPCS chip select defines the characteristics of up to two peripherals. As an example, FLEX_SPI_CRS0 defines the characteristics of the externally decoded peripherals 0 to 1, corresponding to the PCS values 0x0 to 0x1. Consequently, the user has to make sure to connect compatible peripherals on the decoded chip select lines 0 to 1 and 2. The following figure shows this type of implementation.

If the CSAAT bit is used, with or without the DMAC, the mode fault detection for NPCS0 line must be disabled. This is not needed for all other chip select lines since mode fault detection is only on NPCS0.

**Figure 46-72. Chip Select Decoding Application Block Diagram: Single Master/Multiple Slave Implementation**




#### 46.8.3.8 Peripheral Deselection without DMA

During a transfer of more than one data on a Chip Select without the DMA, FLEX_SPI_TDR is loaded by the processor, the TDRE flag rises as soon as the content of FLEX_SPI_TDR is transferred into the internal shift register. When this flag is detected high, FLEX_SPI_TDR can be reloaded. If this reload by the processor occurs before the end of the current transfer, and if the next transfer is performed on the same chip select as the current transfer, the Chip Select is not deasserted between the two transfers. But depending on the application software handling the SPI status register flags (by interrupt or polling method) or servicing other interrupts or other tasks, the processor may not reload FLEX_SPI_TDR in time to keep the chip select active (low). A null DLYBCT value (delay between consecutive transfers) in FLEX_SPI_CSR, gives even less time for the processor to reload FLEX_SPI_TDR. With some SPI slave peripherals, if the chip select line must remain active (low) during a full set of transfers, communication errors can occur.

To facilitate interfacing with such devices, the Chip Select registers [CSR0...CSR1] can be programmed with the Chip Select Active After Transfer (CSAAT) bit to 1. This allows the chip select lines to remain in their current state (low = active) until a transfer to another chip select is required. Even if FLEX_SPI_TDR is not reloaded, the chip select remains active. To de-assert the chip select line at the end of the transfer, the Last Transfer (LASTXFER) bit in FLEX_SPI_CR must be set after writing the last data to transmit into FLEX_SPI_TDR.

#### 46.8.3.9 Peripheral Deselection with DMA

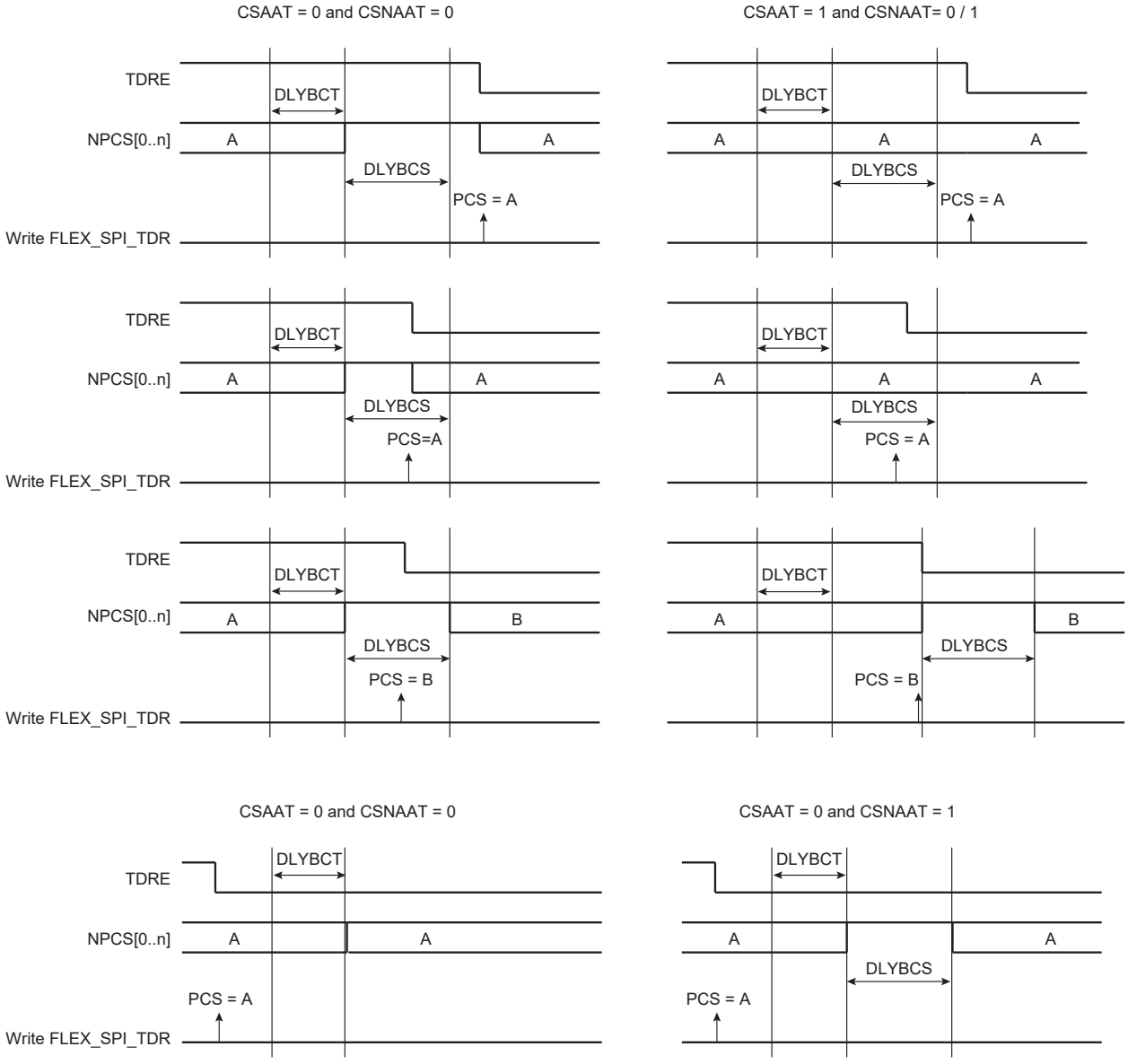
DMA provides faster reloads of FLEX_SPI_TDR compared to software. However, depending on the system activity, it is not guaranteed that FLEX_SPI_TDR is written with the next data before the end of the current transfer. Consequently, a data can be lost by the deassertion of the NPCS line for SPI slave peripherals requiring the chip select line to remain active between two transfers. The only way to guarantee a safe transfer in this case is the use of the CSAAT and LASTXFER bits.

When the CSAAT bit is cleared, the NPCS does not rise in all cases between two transfers on the same peripheral. During a transfer on a Chip Select, the TDRE flag rises as soon as the content of FLEX_SPI_TDR is transferred into the internal shift register. When this flag is detected, FLEX_SPI_TDR can be reloaded. If this reload occurs before the end of the current transfer and if the next transfer is performed on the same chip select as the current transfer, the Chip Select is not deasserted between the two transfers. This can lead to difficulties to interface with some serial peripherals requiring the chip select

to be deasserted after each transfer. To facilitate interfacing with such devices, FLEX_SPI_CSR can be programmed with the Chip Select Not Active After Transfer (CSNAAT) bit to 1. This allows the chip select lines to be deasserted systematically during a time “DLYBCS” (the value of the CSNAAT bit is processed only if the CSAAT bit is cleared for the same chip select).

The following figure shows different peripheral deselection cases and the effect of the CSAAT and CSNAAT bits.

**Figure 46-73. Peripheral Deselection**




### 46.8.3.10 Mode Fault Detection

The SPI has the capability to operate in multi-master environment. Consequently, the NPCS0/NSS line must be monitored. If one of the masters on the SPI bus is currently transmitting, the NPCS0/NSS line is low and the SPI must not transmit a data. A mode fault is detected when the SPI is programmed in Master mode and a low level is driven by an external master on the NPCS0/NSS signal. In multi-master environment, NPCS0, MOSI, MISO and SPCK pins must be configured in open drain (through the PIO controller). When a mode fault is detected, the FLEX_SPI_SR.MODF bit is set until FLEX_SPI_SR is

read and the SPI is automatically disabled until it is re-enabled by writing the FLEX_SPI_CR.SPIEN bit to 1.

By default, the mode fault detection is enabled. The user can disable it by setting the FLEX_SPI_MR.MODFDIS bit.

### 46.8.4 SPI Slave Mode

When operating in Slave mode, the SPI processes data bits on the clock provided on the SPI clock pin (SPCK).

The SPI waits until NSS goes active before receiving the serial clock from an external master. When NSS falls, the clock is validated and the data are loaded in FLEX_SPI_RDR according to the configuration value of the FLEX_SPI_CSR0.BITS field. These bits are processed following a phase and a polarity defined respectively by the FLEX_SPI_CSR0.NCPHA and FLEX_SPI_CSR0.CPOL bits. Note that the BITS field, CPOL bit and NCPHA bit of the other Chip Select registers have no effect when the SPI is programmed in Slave mode.

The bits are shifted out on the MISO line and sampled on the MOSI line.

**Note:** For more information on the BITS field, see also the note below the FLEX_SPI_CSRx register bitmap in section SPI Chip Select Register

When all bits are processed, the received data are transferred in FLEX_SPI_RDR and the RDRF bit rises. If FLEX_SPI_RDR has not been read before new data are received, the Overrun Error bit (OVRES) in FLEX_SPI_SR is set. As long as this flag is set, data are loaded in FLEX_SPI_RDR. The user must read FLEX_SPI_SR to clear the OVRES bit.

When a transfer starts, the data shifted out is the data present in the shift register. If no data has been written in FLEX_SPI_TDR, the last data received is transferred. If no data has been received since the last reset, all bits are transmitted low, as the shift register resets to 0.

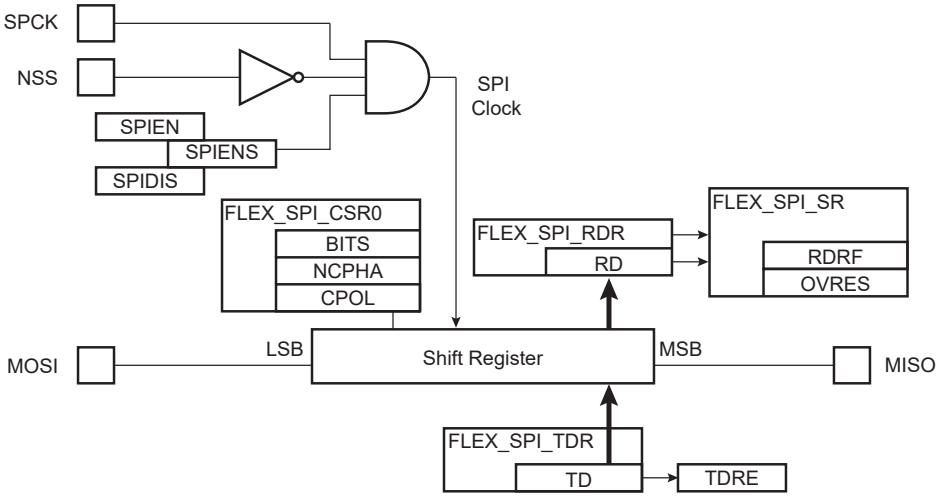
When a first data is written in FLEX_SPI_TDR, it is transferred immediately in the shift register and the TDRE flag rises. If new data is written, it remains in FLEX_SPI_TDR until a transfer occurs, i.e., NSS falls and there is a valid clock on the SPCK pin. When the transfer occurs, the last data written in FLEX_SPI_TDR is transferred in the shift register and the TDRE flag rises. This enables frequent updates of critical variables with single transfers.

Then, a new data is loaded in the shift register from FLEX_SPI_TDR. If no character is ready to be transmitted, i.e., no character has been written in FLEX_SPI_TDR since the last load from FLEX_SPI_TDR to the shift register, FLEX_SPI_TDR is retransmitted. In this case the Underrun Error Status Flag (UNDES) is set in FLEX_SPI_SR.

If NSS rises between two characters, it must be kept high for two MCK clock periods or more and the next SPCK capture edge must not occur less than four MCK periods after NSS rise.

The following figure shows a block diagram of the SPI when operating in Slave mode.

**Figure 46-74. Slave Mode Functional Block Diagram**




### 46.8.5 SPI Comparison Function on Received Character

The comparison is only relevant for SPI Slave mode (MSTR = 0 in FLEX_US_MR).

The effect of a comparison match changes if the system is in Wait or Active mode.

In Wait mode, if asynchronous partial wakeup is enabled, a system wakeup is performed (see section 46.8.6 SPI Asynchronous and Partial Wake-up (SleepWalking)).

In Active mode, the CMP flag in FLEX_SPI_SR is raised. It is set when the received character matches the conditions programmed in the SPI Comparison Register (FLEX_SPI_CMPR). The CMP flag is set as soon as FLEX_SPI_RDR is loaded with the new received character. The CMP flag is cleared by reading FLEX_SPI_SR.

The SPI Comparison Register can be programmed to provide different comparison methods. These are listed below:

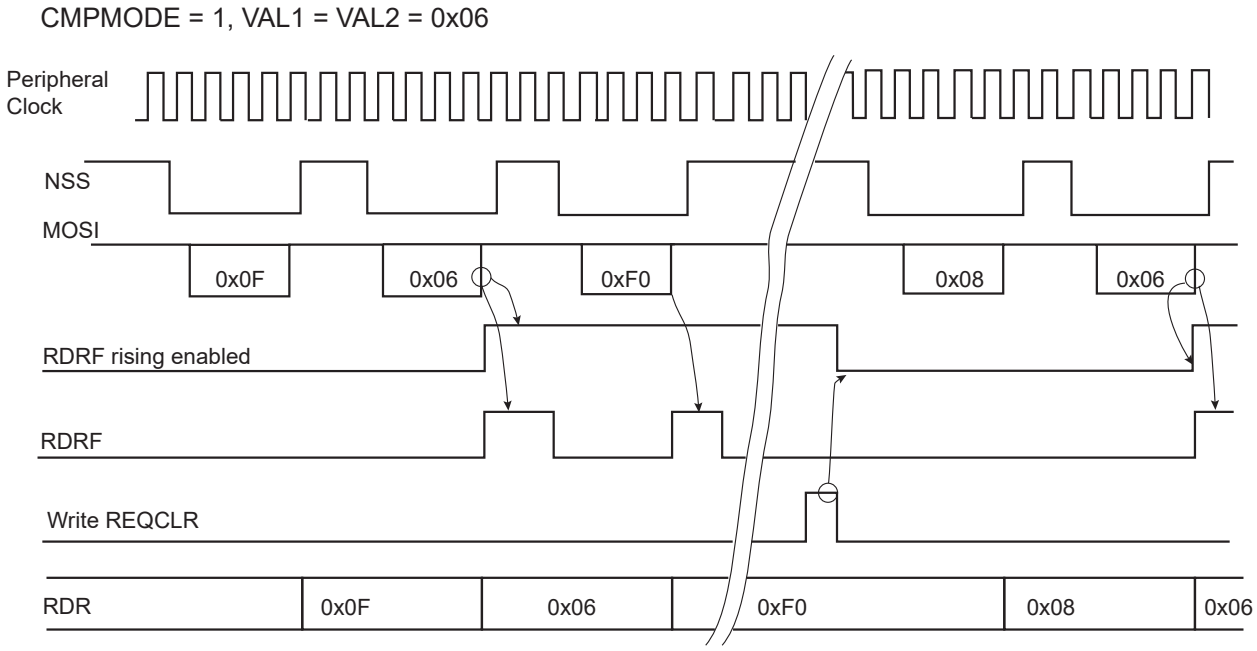
- If VAL1 equals VAL2, then the comparison is performed on a single value and the flag is set to 1 if the received character equals VAL1.
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 sets the CMP flag.
- If VAL1 is strictly higher than VAL2, then the flag CMP is set to 1 if any received character equals VAL1 or VAL2.

When FLEX_SPI_MR.CMPMODE is cleared, all received data is loaded in FLEX_SPI_RDR and the CMP flag provides the status of the comparison result.

By setting the CMPMODE bit, the comparison result triggers the start of FLEX_SPI_RDR loading (see the figure below). The trigger condition exists as soon as the received character value matches the conditions defined by VAL1 and VAL2 in FLEX_SPI_CMPR. The comparison trigger event is restarted by writing a 1 to the FLEX_SPI_CR.REQCLR bit.

The value programmed in VAL1 and VAL2 fields must not exceed the maximum value of the received character (see BITS field in SPI Chip Select Register (FLEX_SPI_CSR)).

**Figure 46-75. Receive Data Register Management**




### 46.8.6 SPI Asynchronous and Partial Wake-up (SleepWalking)

This operating mode is a means of data pre-processing that qualifies an incoming event, thus allowing the SPI to decide whether or not to wake up the system. Asynchronous and partial wake-up is mainly used when the system is in Wait mode (refer to the section “Power Management Controller” (PMC) for further details). It can also be enabled when the system is fully running.

Asynchronous and partial wake-up can be used only when SPI is configured in Slave mode (FLEX_SPI_MR.MSTR is cleared).

The maximum SPI clock (SPCK) frequency that can be provided by the SPI master is bounded by the peripheral clock frequency. The SPCK frequency must be lower than or equal to the peripheral clock. The NSS line must be deasserted by the SPI master between two characters. The NSS deassertion duration time must be greater than or equal to six peripheral clock periods. The time between the assertion of NSS line (falling edge) and the first edge of the SPI clock must be higher than 15 µs.

The FLEX_SPI_RDR register must be read before enabling the asynchronous and partial wake-up.

When asynchronous and partial wake-up is enabled for the SPI (refer to the section “Power Management Controller” (PMC)), the PMC decodes a clock request from the SPI. The request is generated as soon as there is a falling edge on the NSS line as this may indicate the beginning of a frame. If the system is in Wait mode (processor and peripheral clocks switched off), the PMC restarts the fast RC oscillator and provides the clock only to the SPI.

The SPI processes the received frame and compares the received character with VAL1 and VAL2 in FLEX_SPI_CMPR (section FLEX_SPI_CMPR).

The SPI instructs the PMC to disable the peripheral clock if the received character value does not meet the conditions defined by VAL1 and VAL2 fields in FLEX_SPI_CMPR (see Figure 46-76 ).

If the received character value meets the conditions, the SPI instructs the PMC to exit the system from Wait mode (see Figure 46-77).

The VAL1 and VAL2 fields can be programmed to provide different comparison methods and thus matching conditions.

- If VAL1 equals VAL2, then the comparison is performed on a single value and the wake-up is triggered if the received character equals VAL1.
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 wakes up the system.
- If VAL1 is strictly higher than VAL2, the wake-up is triggered if any received character equals VAL1 or VAL2.
- If VAL1 = 0 and VAL2 = 65535, the wake-up is triggered as soon as a character is received.

If the processor and peripherals are running, the SPI can be configured in Asynchronous and Partial Wake-up mode by enabling the PMC_SLPWK_ER (refer to the section “Power Management Controller” (PMC)). When activity is detected on the receive line, the SPI requests the clock from the PMC and the comparison is performed. If there is a comparison match, the SPI continues to request the clock. If there is no match, the clock is switched off for the SPI only, until a new activity is detected.

The CMPMODE configuration has no effect when Asynchronous and Partial Wake-up mode is enabled for the SPI (refer to PMC_SLPWK_ER in the section “Power Management Controller” (PMC)).

When the system is in Active mode and the SPI enters Asynchronous and Partial Wake-up mode, the flag RDRF must be programmed as the unique source of the SPI interrupt.

When the system exits Wait mode as the result of a matching condition, the RDRF flag is used to determine if the SPI is the source for the exit from Wait mode.

**Figure 46-76. Asynchronous Wake-up Use Case Example**

Case with VAL1 = VAL2 = 0x55

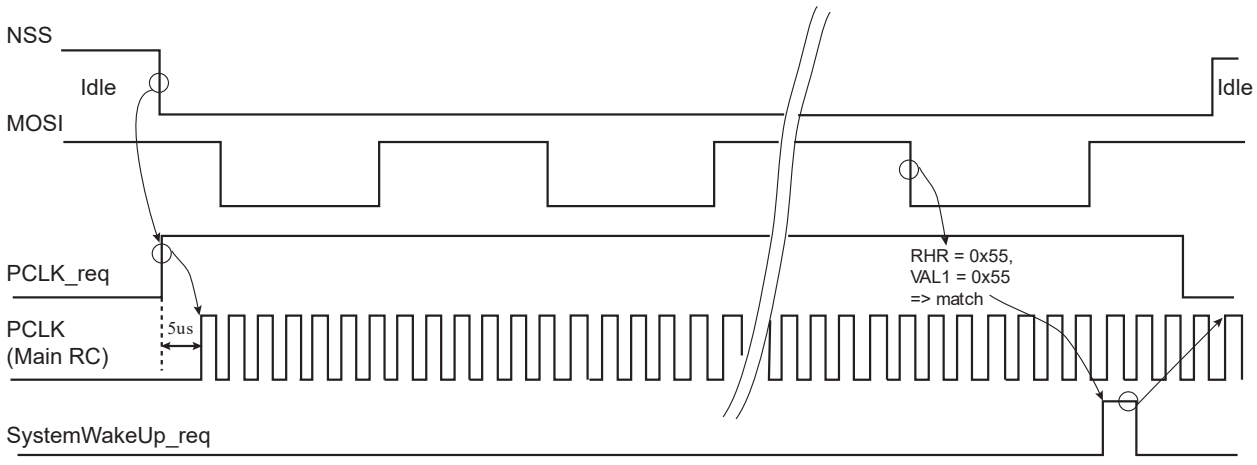
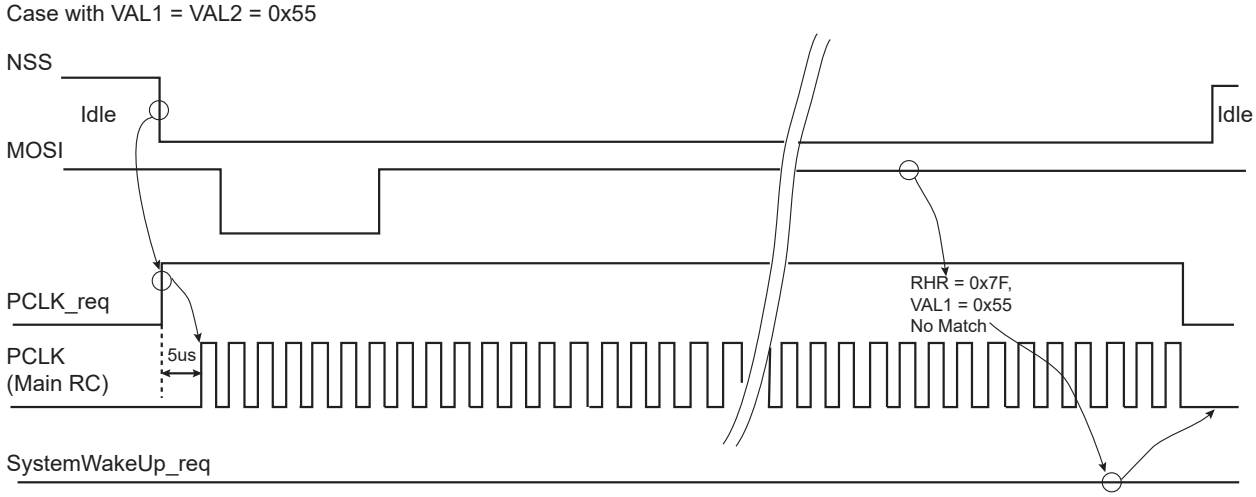


**Figure 46-77. Asynchronous Event Generating Only Partial Wake-up**




### 46.8.7 SPI FIFOs

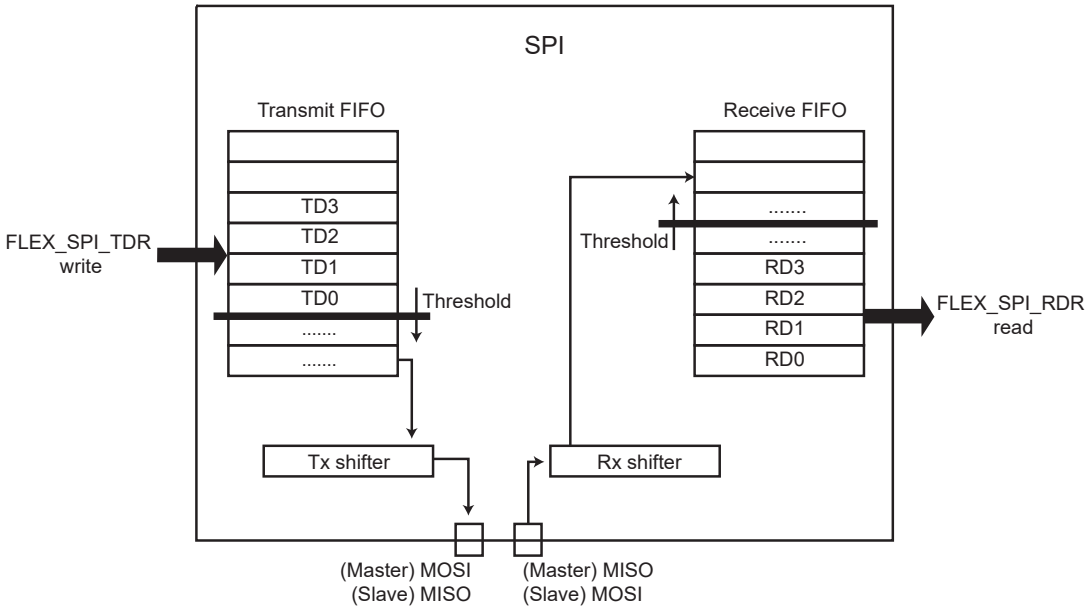
#### 46.8.7.1 Overview

The SPI includes two FIFOs which can be enabled/disabled using the FLEX_SPI_CR.FIFOEN/FIFODIS. The SPI module must be disabled before enabling or disabling the SPI FIFOs (FLEX_SPI_CR.SPIDIS).

Writing FLEX_SPI_CR.FIFOEN to '1' enables a FIFO_DEPTH-data Transmit FIFO and a FIFO_DEPTH-data Receive FIFO.

It is possible to write or to read single or multiple data in the same access to FLEX_SPI_TDR/RDR. See sections SPI Single Data Mode and SPI Multiple Data Mode.

**Figure 46-78. FIFOs Block Diagram**




#### 46.8.7.2 Sending Data with FIFO Enabled

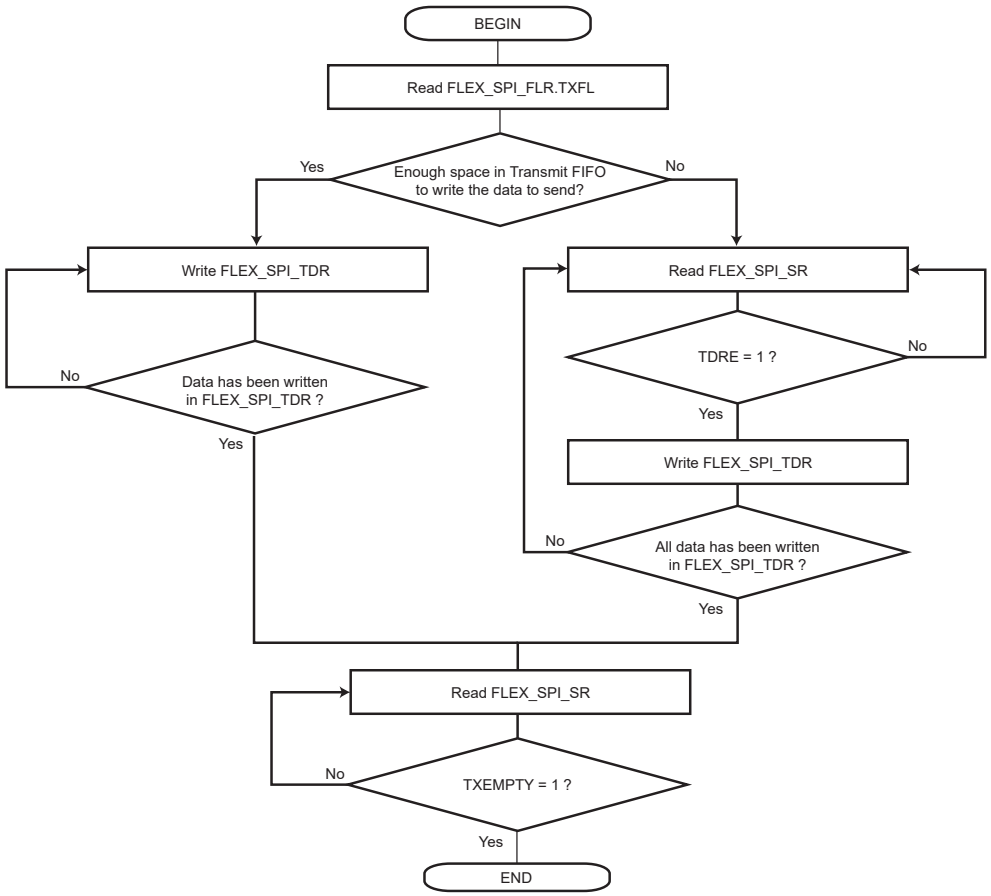
When the Transmit FIFO is enabled, write access to FLEX_SPI_TDR loads the Transmit FIFO.

The FIFO level is provided in FLEX_SPI_FLR.TXFL. If the FIFO can accept the number of data to be transmitted, there is no need to monitor FLEX_SPI_SR.TDRE and the data can be successively written in FLEX_SPI_TDR.

If the FIFO cannot accept the data due to insufficient space, wait for the TDRE flag to be set before writing the data in FLEX_SPI_TDR.

When the space in the FIFO allows only a portion of the data to be written, the TDRE flag must be monitored before writing the remaining data.

**Figure 46-79. Sending Data with FIFO Enabled**

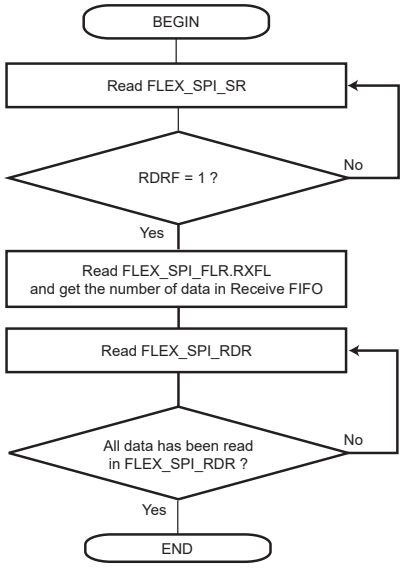



### 46.8.7.3 Receiving Data with FIFO Enabled

When the Receive FIFO is enabled, FLEX_SPI_RDR access reads the FIFO.

When data are present in the Receive FIFO (RDRF flag set to ‘1’), the exact number of data can be checked with FLEX_SPI_FLR.RXFL. All the data can be read successively in FLEX_SPI_RDR without checking the RDRF flag between each access.

**Figure 46-80. Receiving Data with FIFO Enabled**




#### 46.8.7.4 Clearing/Flushing FIFOs

Each FIFO can be cleared/flushed using FLEX_SPI_CR.TXFCLR/RXFCLR.

#### 46.8.7.5 TXEMPTY, TDRE and RDRF Behavior

FLEX_SPI_SR.TXEMPTY, FLEX_SPI_SR.TDRE and FLEX_SPI_SR.RDRF flags display a specific behavior when FIFOs are enabled.

The TXEMPTY flag is cleared as long as there are characters in the Transmit FIFO or in the internal shift register. TXEMPTY is set when there are no characters in the Transmit FIFO and in the internal shift register.

TDRE indicates if a data can be written in the Transmit FIFO. Thus the TDRE flag is set as long as the Transmit FIFO can accept new data. See figure TDRE in Single Data Mode and TXRDYM = 0.

RDRF indicates if an unread data is present in the Receive FIFO. Thus the RDRF flag is set as soon as one unread data is in the Receive FIFO. See figure RDRF in Single Data Mode and RXRDYM = 0.

TDRE and RDRF behavior can be modified using the TXRDYM and RXRDYM fields in the SPI FIFO Mode Register (FLEX_SPI_FMR) to reduce the number of accesses to FLEX_SPI_TDR/RDR. However, for some configurations, the following constraints apply:

- When the Variable Peripheral Select mode is used (FLEX_SPI_MR.PS=1), TXRDYM/RXRDYM must be cleared.
- In Master mode (FLEX_SPI_MR.MSTR=1), RXRDYM must be cleared.

As an example, in Master mode, the Transmit FIFO can be loaded with multiple data in the same access by configuring TXRDYM>0.

See SPI FIFO Mode Register (FLEX_SPI_FMR) for the FIFO configuration.

**Figure 46-81. TDRE in Single Data Mode and TXRDYM = 0**

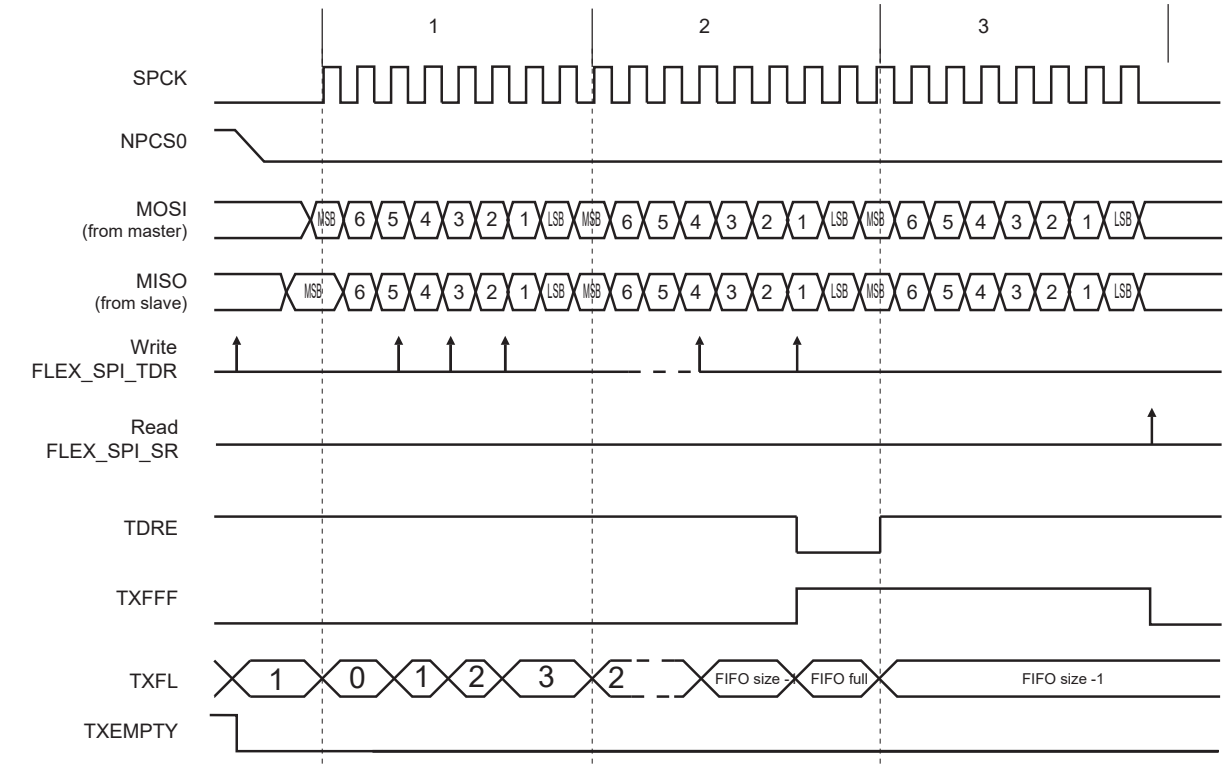
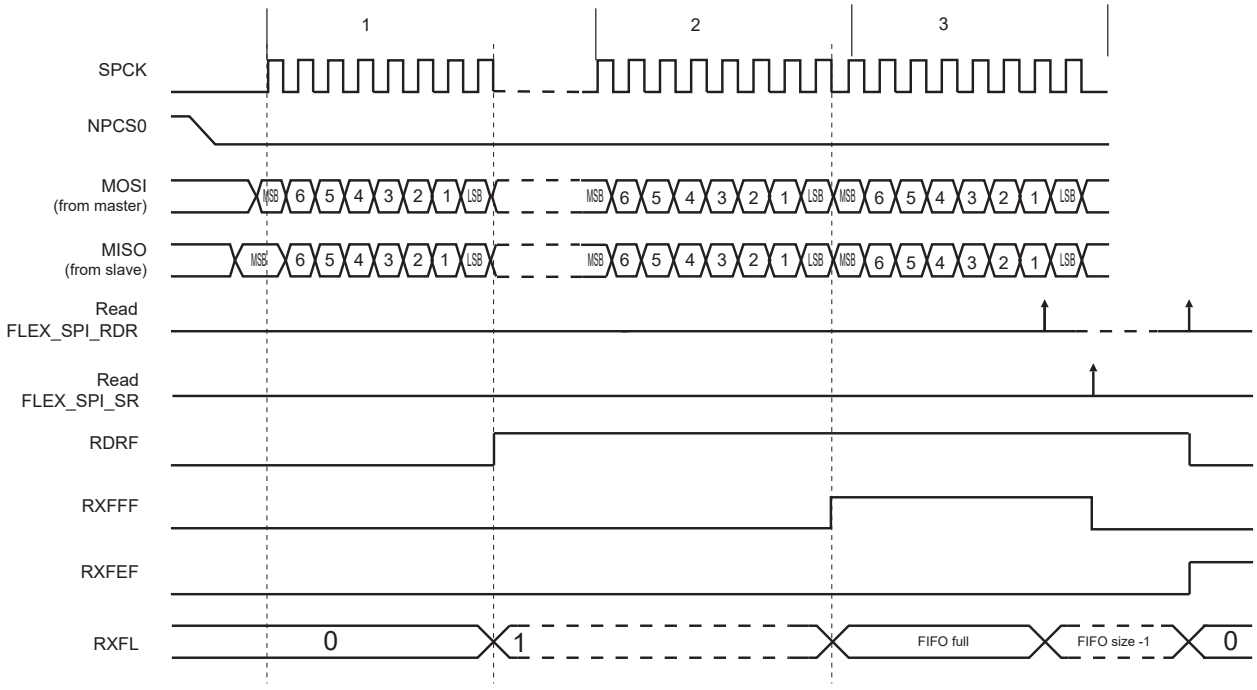



**Figure 46-82. RDRF in Single Data Mode and RXRDYM = 0**




### 46.8.7.6 SPI Single Data Mode

In Single Data mode, only one data is written every time FLEX_SPI_TDR is accessed, and only one data is read every time FLEX_SPI_RDR is accessed.

When FLEX_SPI_FMR.TXRDYM = 0, the Transmit FIFO operates in Single Data mode.

When FLEX_SPI_FMR.RXRDYM = 0, the Receive FIFO operates in Single Data mode.

If Master mode is used (FLEX_SPI_MR.MSTR=1), the Receive FIFO must operate in Single Data mode.

If Variable Peripheral Select mode is used (FLEX_SPI_MR.PS=1), the Transmit FIFO must operate in Single Data mode.

See sections SPI Transmit Data Register and SPI Receive Data Register.

#### 46.8.7.6.1 DMAC

When FIFOs operate in Single Data mode, the DMAC transfer type must be configured either in bytes, halfwords or words depending on FLEX_SPI_MR.PS bit value and FLEX_SPI_CSRx.BITS field value.

The same applies when FIFOs are disabled.

### 46.8.7.7 SPI Multiple Data Mode

Multiple Data mode minimizes the number of accesses by concatenating the data to send/read in one access.

When FLEX_SPI_FMR.TXRDYM > 0, the Transmit FIFO operates in Multiple Data mode.

When FLEX_SPI_FMR.RXRDYM > 0, the Receive FIFO operates in Multiple Data mode.

Multiple data can be read from the Receive FIFO only in Slave mode (FLEX_SPI_MR.MSTR=0).

The Transmit FIFO can be loaded with multiple data in the same access by configuring TXRDYM>0 and when FLEX_SPI_MR.PS=0.

In Multiple Data mode, up to two data can be written in one FLEX_SPI_TDR write access. It is also possible to read up to four data in one FLEX_SPI_RDR access if FLEX_SPI_CSRx.BITS is configured to ‘0’ (8-bit data size) and up to two data if FLEX_SPI_CSRx.BITS is configured to a value other than ‘0’ (more than 8-bit data size).

The number of data to write/read is defined by the size of the register access. If the access is a byte-size register access, only one data is written/read. If the access is a halfword size register access, then up to two data are read and only one data is written. Lastly, if the access is a word-size register access, then up to four data are read and up to two data are written.

Written/read data are always right-aligned, as described in sections SPI Receive Data Register (FIFO Multiple Data, 8-bit), SPI Receive Data Register (FIFO Multiple Data, 16-bit) and SPI Transmit Data Register (FIFO Multiple Data, 8- to 16-bit).

As an example, if the Transmit FIFO is empty and there are six data to send, either of the following write accesses may be performed:

- six FLEX_SPI_TDR-byte write accesses
- three FLEX_SPI_TDR-halfword write accesses

With a Receive FIFO containing six data, any of the following read accesses may be performed:

- six FLEX_SPI_RDR-byte read accesses
- three FLEX_SPI_RDR-halfword read accesses
- one FLEX_SPI_RDR-word read access and one FLEX_SPI_RDR-halfword read access

#### 46.8.7.7.1 TDRE and RDRF Configuration

In Multiple Data mode, it is possible to write one or more data in the same FLEX_SPI_TDR/RDR access. The TDRE flag indicates if one or more data can be written in the FIFO depending on the configuration of FLEX_SPI_FMR.TXRDYM/RXRDYM.

As an example, if two data are written each time in FLEX_SPI_TDR, it is useful to configure the TXRDYM field to the value ‘1’ so that the TDRE flag is at ‘1’ only when at least two data can be written in the Transmit FIFO.

Similarly, if four data are read each time in FLEX_SPI_RDR, it is useful to configure the RXRDYM field to the value ‘2’ so that the RDRF flag is at ‘1’ only when at least four unread data are in the Receive FIFO.

#### 46.8.7.7.2 DMAC

It is mandatory to configure DMAC channel size (byte, halfword or word) according to FLEX_SPI_FMR.TXRDYM/RXRDYM configuration. See section SPI Multiple Data Mode for constraints.

### 46.8.7.8 FIFO Pointer Error

A FIFO overflow is reported in FLEX_SPI_SR.

If the Transmit FIFO is full and a write access is performed on FLEX_SPI_TDR, it generates a Transmit FIFO pointer error and sets FLEX_SPI_SR.TXFPTEF.

In Multiple Data mode, if the number of data written in FLEX_SPI_TDR (according to the register access size) is greater than the free space in the Transmit FIFO, a Transmit FIFO pointer error is generated and FLEX_SPI_SR.TXFPTEF is set.

A FIFO underflow is reported in FLEX_SPI_SR.

In Multiple Data mode, if the number of data read in FLEX_SPI_RDR (according to the register access size) is greater than the number of unread data in the Receive FIFO, a Receive FIFO pointer error is generated and FLEX_SPI_SR.RXFPTEF is set.

No pointer error occurs if the FIFO state/level is checked before writing/reading in FLEX_SPI_TDR/SPI_RDR. The FIFO state/level can be checked either with TXRDY, RXRDY, TXFL or RXFL. When a pointer error occurs, other FIFO flags may not behave as expected; their states should be ignored.

If a pointer error occurs, a software reset must be performed using FLEX_SPI_CR.SWRST (configuration will be lost).

### 46.8.7.9 FIFO Thresholds

Each Transmit and Receive FIFO includes a threshold feature used to set a flag and an interrupt when a FIFO threshold is crossed. Thresholds are defined as a number of data in the FIFO, and the FIFO state (TXFL or RXFL) represents the number of data currently in the FIFO.

The Transmit FIFO threshold can be set using the field FLEX_SPI_FMR.TXFTHRES. Each time the Transmit FIFO level goes from ‘above threshold’ to ‘equal to or below threshold’, the flag FLEX_SPI_SR.TXFTHF is set. The application is warned that the Transmit FIFO has reached the defined threshold and that it can be reloaded.

The Receive FIFO threshold can be set using the field FLEX_SPI_FMR.RXFTHRES. Each time the Receive FIFO level goes from ‘below threshold’ to ‘equal to or above threshold’, the flag FLEX_SPI_SR.RXFTHF is set. The application is warned that the Receive FIFO has reached the defined threshold and that it can be read to prevent an underflow.

The TXFTHF and RXFTHF flags can be configured to generate an interrupt using FLEX_SPI_IER and FLEX_SPI_IDR.

### 46.8.7.10 FIFO Flags

FIFOs come with a set of flags which can be configured to generate interrupts through FLEX_SPI_IER and FLEX_SPI_IDR.

FIFO flags state can be read in FLEX_SPI_SR. They are cleared when FLEX_SPI_SR is read.

### 46.8.8 SPI Register Write Protection

The FLEXCOM operating mode (FLEX_MR.OPMODE) must be set to FLEX_MR_OPMODE_SPI to enable access to the write protection registers.

To prevent any single software error from corrupting SPI behavior, certain registers in the address space can be write-protected by setting the WPEN (Write Protection Enable), WPITEN (Write Protection Interrupt Enable), and/or WPCREN (Write Protection Control Enable) bits in the SPI Write Protection Mode Register (FLEX_SPI_WPMR).

If a write access to a write-protected register is detected, the Write Protection Violation Status (WPVS) flag in the SPI Write Protection Status Register (FLEX_SPI_WPSR) is set and the Write Protection Violation Source (WPVSRC) field indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading FLEX_SPI_WPSR.

The following registers can be write-protected when WPEN is set:

- SPI Mode Register
- SPI Chip Select Register
- SPI Comparison Register

The following registers can be write-protected when WPITEN is set:

- SPI Interrupt Enable Register
- SPI Interrupt Disable Register

The following register can be write-protected when WPCREN is set:

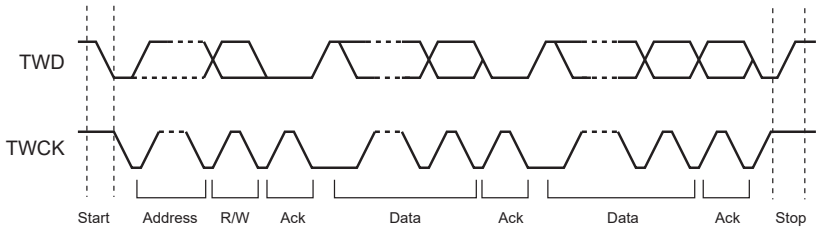
- SPI Control Register

## 46.9 TWI Functional Description

### 46.9.1 Transfer Format

The data put on the TWD line must be 8 bits long. Data are transferred MSB first; each byte must be followed by an acknowledgement. The number of bytes per transfer is unlimited (see figure below).

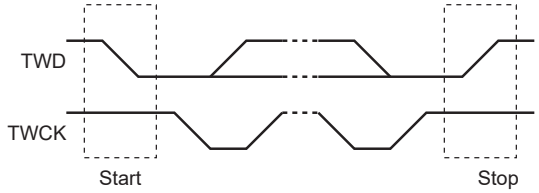
**Figure 46-83. Transfer Format**




Each transfer begins with a START condition and terminates with a STOP condition (see figure below).

- A high-to-low transition on the TWD line while TWCK is high defines the START condition.
- A low-to-high transition on the TWD line while TWCK is high defines a STOP condition.

**Figure 46-84. START and STOP Conditions**




### 46.9.2 Modes of Operation

The TWI has different modes of operation:

- Master Transmitter mode (Standard and Fast modes only)
- Master Receiver mode (Standard and Fast modes only)
- Multi-master Transmitter mode (Standard and Fast modes only)
- Multi-master Receiver mode (Standard and Fast modes only)
- Slave Transmitter mode (Standard, Fast and High-speed modes)
- Slave Receiver mode (Standard, Fast and High-speed modes)

These modes are described in the following sections.

### 46.9.3 Master Mode

#### 46.9.3.1 Definition

The master is the device that starts a transfer, generates a clock and stops it. This operating mode is not available if High-speed mode is selected.

#### 46.9.3.2 Programming Master Mode

The following fields must be programmed before entering Master mode:

1. DADR (+ IADRSZ + IADR if a 10-bit device is addressed): The device address is used to access slave devices in Read or Write mode.
2. CWGR + CKDIV + CHDIV + CLDIV: Clock waveform.
3. SVDIS: Disables Slave mode.
4. MSEN: Enables Master mode.
   **Note:** If the TWI is already in Master mode, the device address (DADR) can be configured without disabling the Master mode.

#### 46.9.3.3 Transfer Speed/Bit Rate

The TWI speed is defined in FLEX_TWI_CWGR. The TWI bit rate can be based either on the peripheral clock if the BRSRCCLK bit value is 0 or on a programmable clock source provided by the GCLK if the BRSRCCLK bit value is 1.

If BRSRCCLK = 1, the bit rate is independent of the processor/peripheral clock and thus processor/peripheral clock frequency can be changed without affecting the TWI transfer rate.

The GCLK frequency must be at least three times lower than the peripheral clock frequency.

#### 46.9.3.4 Master Transmitter Mode

This operating mode is not available if High-speed mode is selected.

After the master initiates a START condition when writing into the Transmit Holding register FLEX_TWI_THR, it sends a 7-bit slave address, configured in the Master Mode Register (DADR in FLEX_TWI_MMR), to notify the slave device. The bit following the slave address indicates the transfer direction, 0 in this case (FLEX_TWI_MMR.MREAD = 0).

The TWI transfers require the slave to acknowledge each received byte. During the acknowledge clock pulse (ninth pulse), the master releases the data line (HIGH), enabling the slave to pull it down in order to generate the acknowledge. If the slave does not acknowledge the byte, then the Not Acknowledge flag (NACK) is set in the TWI Status Register (FLEX_TWI_SR) of the master and a STOP condition is sent. Alternatively, if the FLEX_TWI_MMR.NOAP bit is set, no stop condition will be sent and a START or STOP condition must be triggered manually through the FLEX_TWI_CR.START or FLEX_TWI_CR.STOP bit once the software is ready for the transmission of the condition. The NACK flag must be cleared by reading the TWI Status Register (FLEX_TWI_SR) before the next write into the TWI Transmit Holding Register (FLEX_TWI_THR). As with the other status bits, an interrupt can be generated if enabled in the interrupt enable Register (FLEX_TWI_IER). If the slave acknowledges the byte, the data written in FLEX_TWI_THR is then shifted in the internal shifter and transferred. When an acknowledge is detected, the TXRDY bit is set until a new write in FLEX_TWI_THR.

TXRDY is used as transmit ready for the DMA transmit channel.

**Note:** To clear the TXRDY flag in Master mode, write the FLEX_TWI_CR.MSDIS bit to 1, then write the FLEX_TWI_CR.MSEN bit to 1.

While no new data is written in FLEX_TWI_THR, the serial clock line is tied low. When new data is written in FLEX_TWI_THR, the SCL is released and the data is sent. To generate a STOP event, the STOP command must be performed by writing in the STOP field of the TWI Control Register (FLEX_TWI_CR).

After a master write transfer, the Serial Clock line is stretched (tied low) while no new data is written in FLEX_TWI_THR or until a STOP command is performed.

See the following figures.

**Figure 46-85. Master Write with One Data Byte**

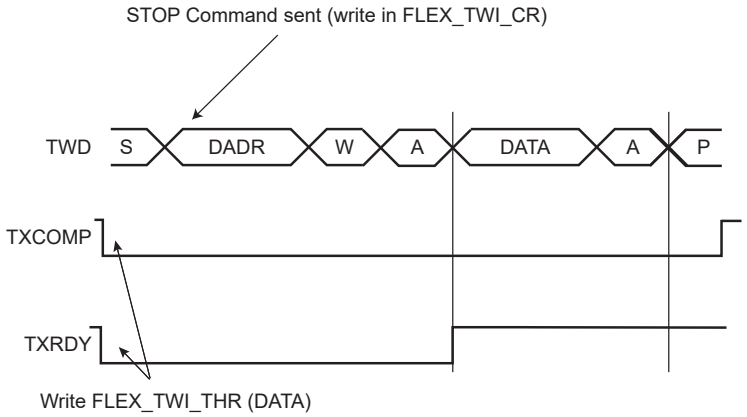


**Figure 46-86. Master Write with Multiple Data Bytes**

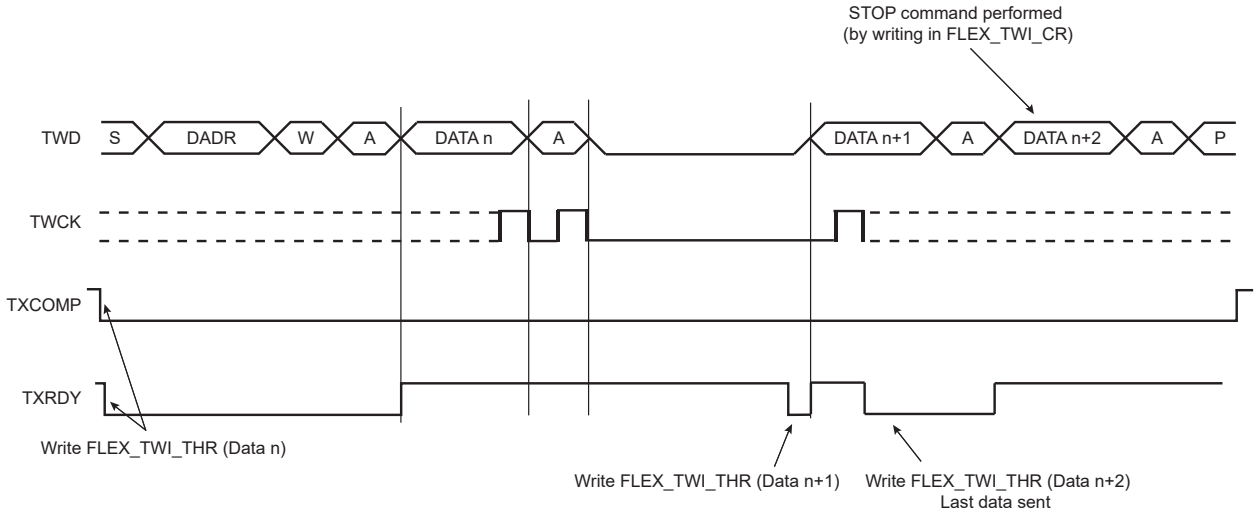
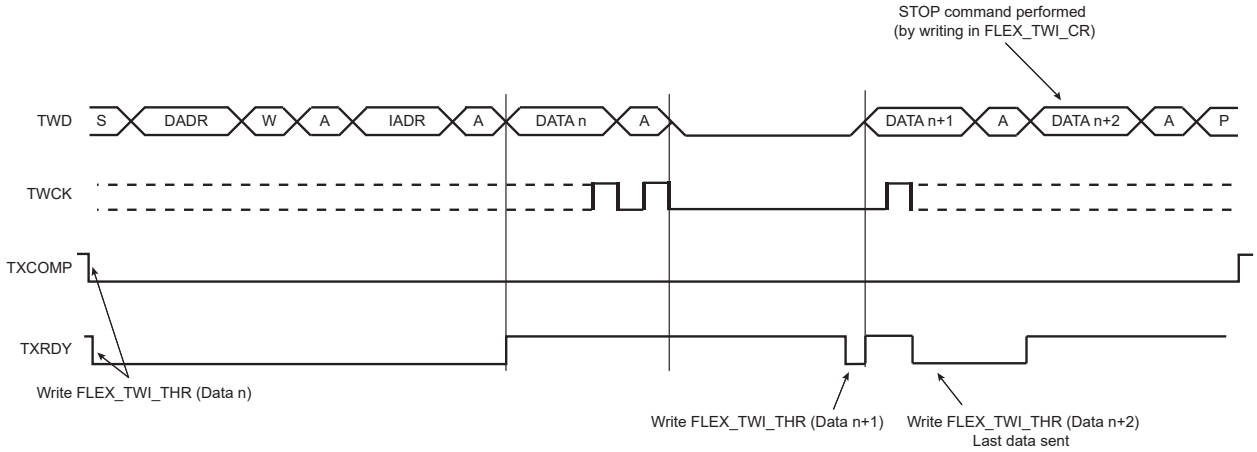



**Figure 46-87. Master Write with One Byte Internal Address and Multiple Data Bytes**




#### 46.9.3.5 Master Receiver Mode

Master Receiver mode is not available if High-speed mode is selected.

The read sequence begins by setting the START bit. After the start condition has been sent, the master sends a 7-bit slave address to notify the slave device. The bit following the slave address indicates the transfer direction, 1 in this case (FLEX_TWI_MMR.MREAD = 1). During the acknowledge clock pulse (9th pulse), the master releases the data line (HIGH), enabling the slave to pull it down in order to generate the acknowledge. The master polls the data line during this clock pulse and sets the FLEX_TWI_SR.NACK bit if the slave does not acknowledge the byte.

If an acknowledge is received, the master is then ready to receive data from the slave. After data has been received, the master sends an acknowledge condition to notify the slave that the data has been received except for the last data (see figure "Master Read with One Data Byte" below). When the FLEX_TWI_SR.RXRDY bit is set, a character has been received in the Receive Holding Register (FLEX_TWI_RHR). The RXRDY bit is reset when reading FLEX_TWI_RHR.

When a single data byte read is performed, with or without internal address (IADR), the START and STOP bits must be set at the same time. See figure "Master Read with One Data Byte" below. When a multiple data byte read is performed, with or without internal address (IADR), the STOP bit must be set after the next-to-last data received (same condition applies for START bit to generate a repeated start).

See figure "Master Read with Multiple Data Bytes" below. For internal address usage, see section Internal Address.

If FLEX_TWI_RHR is full (RXRDY high) and the master is receiving data, the serial clock line will be tied low before receiving the last bit of the data and until FLEX_TWI_RHR is read. Once FLEX_TWI_RHR is read, the master will stop stretching the serial clock line and end the data reception. See figure "Master Read Clock Stretching with Multiple Data Bytes" below.

**WARNING** When receiving multiple bytes in Master Read mode, if the next-to-last access is not read (the RXRDY flag remains high), the last access will not be completed until FLEX_TWI_RHR is read. The last access stops on the next-to-last bit (clock stretching). When FLEX_TWI_RHR is read there is only half a bit period to send the STOP bit (or START bit) command, else another read access might occur (spurious access).

A possible workaround is to set the STOP bit (or START bit) before reading FLEX_TWI_RHR on the next-to-last access (within IT handler).

**Figure 46-88. Master Read with One Data Byte**

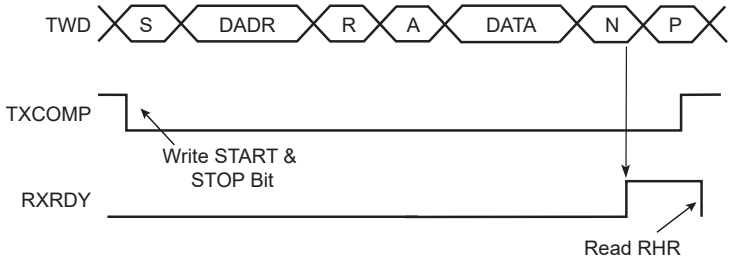



**Figure 46-89. Master Read with Multiple Data Bytes**

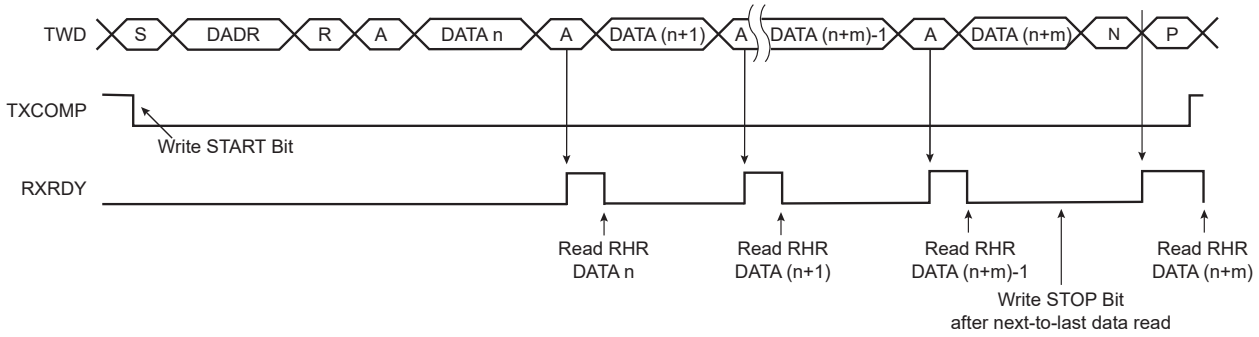
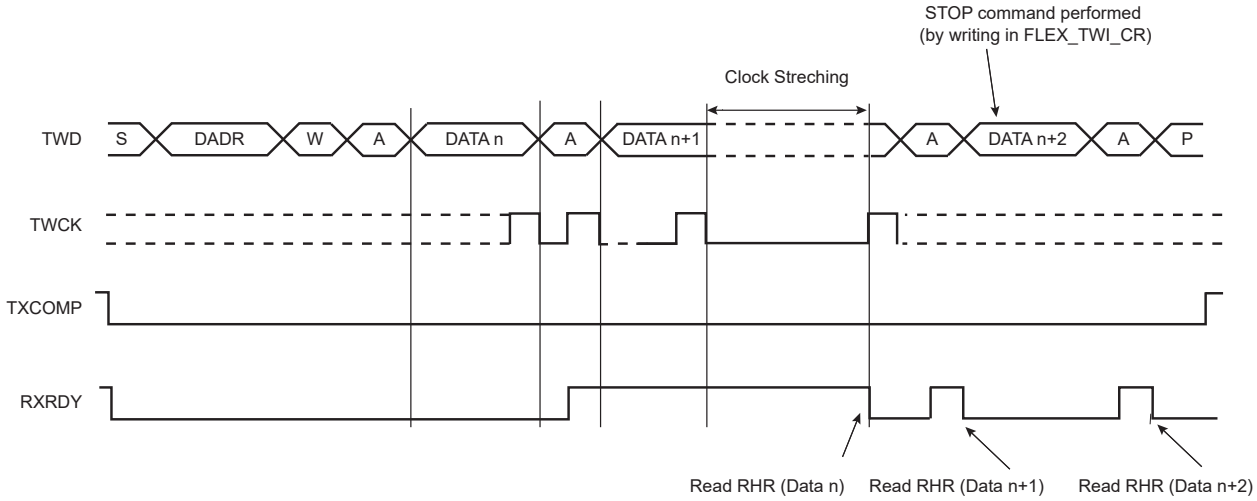


**Figure 46-90. Master Read Clock Stretching with Multiple Data Bytes**




RXRDY is used as receive ready trigger event for the DMA receive channel.

#### 46.9.3.6 Internal Address

The TWI interface can perform transfers with 7-bit slave address devices and with 10-bit slave address devices.

##### 46.9.3.6.1 7-bit Slave Addressing

When addressing 7-bit slave devices, the internal address bytes are used to perform random address (read or write) accesses to reach one or more data bytes, e.g., within a memory page location in a serial memory. When performing read operations with an internal address, the TWI performs a write operation to set the internal address into the slave device, and then switch to Master Receiver mode. Note that the second start condition (after sending the IADR) is sometimes called “repeated start” (Sr) in $I^2C$ fully-compatible devices. See figure Master Read with One, Two or Three Bytes Internal Address and One Data Byte.

See figures Master Write with One, Two or Three Bytes Internal Address and One Data Byte and Internal Address Usage for the master write operation with internal address.

The three internal address bytes are configurable through the Master Mode Register (FLEX_TWI_MMR).

If the slave device supports only a 7-bit address, i.e., no internal address, IADRSZ must be configured to 0.

The abbreviations listed below are used in the following figures:

| | |
|---|---|
| S | Start |
| Sr | Repeated Start |
| P | Stop |
| W | Write |
| R | Read |
| A | Acknowledge |
| N | Not Acknowledge |
| DADR | Device Address |

| | |
|---|---|
| IADR | Internal Address |

**Figure 46-91. Master Write with One, Two or Three Bytes Internal Address and One Data Byte**

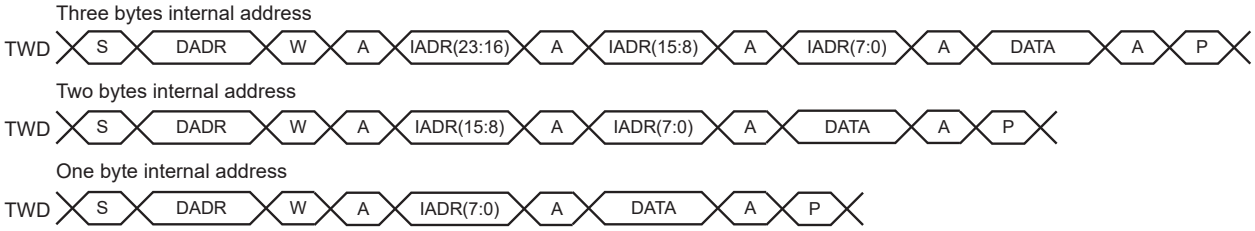
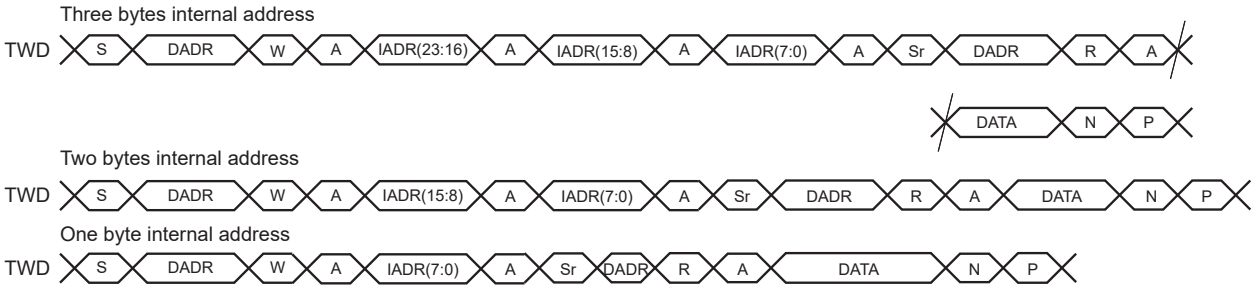



**Figure 46-92. Master Read with One, Two or Three Bytes Internal Address and One Data Byte**




#### 46.9.3.6.2 10-bit Slave Addressing

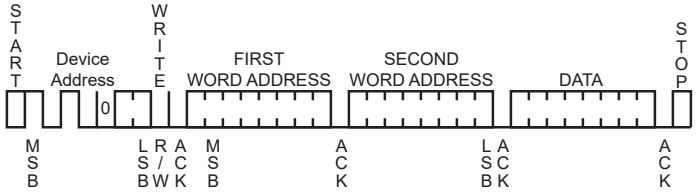
For a slave address higher than seven bits, the user must configure the address size (IADRSZ) and set the other slave address bits in the Internal Address Register (FLEX_TWI_IADR). The two remaining internal address bytes, IADR[15:8] and IADR[23:16], can be used the same way as in 7-bit slave addressing.

Example: Address a 10-bit device (10-bit device address is b1 b2 b3 b4 b5 b6 b7 b8 b9 b10)

1. Program IADRSZ = 1
2. Program DADR with 1 1 1 1 0 b1 b2 (b1 is the MSB of the 10-bit address, b2, etc.)
3. Program FLEX_TWI_IADR with b3 b4 b5 b6 b7 b8 b9 b10 (b10 is the LSB of the 10-bit address)

The following figure shows a byte write to a TWI EEPROM. This demonstrates the use of internal addresses to access the device.

**Figure 46-93. Internal Address Usage**




### 46.9.3.7 Repeated Start

In addition to Internal Address mode, repeated start (Sr) can be generated manually by writing the START bit at the end of a transfer instead of the STOP bit. In such case the parameters of the next transfer (direction, SADR, etc.) will need to be set before writing the START bit at the end of the previous transfer.

See section Read/Write Flowcharts.

### 46.9.3.8 Bus Clear Command

The TWI interface can perform a Bus Clear Command:

1. Configure the Master mode (DADR, CKDIV, etc).
2. Start the transfer by setting the FLEX_TWI_CR.CLEAR bit.

**Note:** If an alternative command is used (ACMEN bit = 1), the DATAL field must be cleared.

#### 46.9.3.9 SMBus Mode

SMBus mode is enabled when the FLEX_TWI_CR.SMBEN bit is written to one. SMBus mode operation is similar to I²C operation with the following exceptions:

1. Only 7-bit addressing can be used.
2. The SMBus standard describes a set of timeout values to ensure progress and throughput on the bus. These timeout values must be programmed into FLEX_TWI_SMBTR.
3. Transmissions can optionally include a CRC byte, called Packet Error Check (PEC).
4. A set of addresses has been reserved for protocol handling, such as alert response address (ARA) and host header (HH) address. Address matching on these addresses can be enabled by configuring FLEX_TWI_CR appropriately.

##### 46.9.3.9.1 Packet Error Checking

Each SMBus transfer can optionally end with a CRC byte, called the PEC byte. Writing the FLEX_TWI_CR.PECEN bit to one enables automatic PEC handling in the current transfer. Transfers with and without PEC can freely be intermixed in the same system, since some slaves may not support PEC. The PEC LFSR is always updated on every bit transmitted or received, so that PEC handling on combined transfers will be correct.

In Master Transmitter mode, the master calculates a PEC value and transmits it to the slave after all data bytes have been transmitted. Upon reception of this PEC byte, the slave will compare it to the PEC value it has computed itself. If the values match, the data was received correctly, and the slave will return an ACK to the master. If the PEC values differ, data was corrupted, and the slave will return a NACK value. Some slaves may not be able to check the received PEC in time to return a NACK if an error occurred. In this case, the slave should always return an ACK after the PEC byte, and some other mechanism must be implemented to verify that the transmission was received correctly.

In Master Receiver mode, the slave calculates a PEC value and transmits it to the master after all data bytes have been transmitted. Upon reception of this PEC byte, the master will compare it to the PEC value it has computed itself. If the values match, the data was received correctly. If the PEC values differ, data was corrupted, and the FLEX_TWI_SR.PECERR bit is set. In Master Receiver mode, the PEC byte is always followed by a NACK transmitted by the master, since it is the last byte in the transfer.

In combined transfers, the PECRQ bit should only be set in the last of the combined transfers. If Alternative Command mode is enabled, only the NPEC bit should be set.

Consider the following transfer:

S, ADR+W, COMMAND_BYTE, ACK, SR, ADR+R, DATA_BYTE, ACK, PEC_BYTE, NACK, P

See section Read/Write Flowcharts for detailed flowcharts.

##### 46.9.3.9.2 Timeouts

The FLEX_TWI_SMBTR.TLOWS/TLOWM fields configure the SMBus timeout values. If a timeout occurs, the master transmits a STOP condition and leaves the bus. Furthermore, the FLEX_TWI_SR.TOUT bit is set.

#### 46.9.3.10 SMBus Quick Command (Master Mode Only)

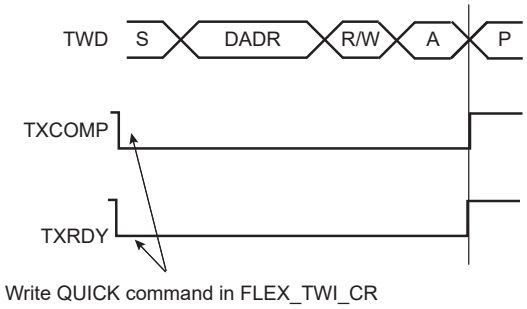
The TWI interface can perform a quick command:

1. Configure the Master mode (DADR, CKDIV, etc).

2. Write the FLEX_TWI_MMR.MREAD bit at the value of the one-bit command to be sent.
3. Start the transfer by setting the FLEX_TWI_CR.QUICK bit.

**Note:** If an alternative command is used (ACMEN bit = 1), the DATAL field must be cleared.

**Figure 46-94. SMBus Quick Command**




#### 46.9.3.11 Alternative Command

Another way to configure the transfer is to enable the Alternative Command mode with the ACMEN bit of the TWI Control Register.

In this mode, the transfer is configured through the TWI Alternative Command Register. It is possible to define a simple read or write transfer or a combined transfer with a repeated start.

In order to set a simple transfer, the DATAL field and the DIR field of the TWI Alternative Command Register must be filled accordingly and the NDATAL field must be cleared. To begin the transfer, either set the START bit in the TWI Control Register in case of a read transfer, or write the TWI Transmit Holding Register in case of a write transfer.

For a combined transfer linked by a repeated start, the NDATAL field must be filled with the length of the second transfer and NDIR with the corresponding direction.

The PEC and NPEC bits are used to set a PEC field. In the case of a single transfer with PEC, the PEC bit must be set. In the case of a combined transfer, the NPEC bit must be set.

**Note:** If the Alternative Command mode is used, the TWIHS_MMR.IADRSZ field must be set to 0.

See Read/Write Flowcharts for detailed flowcharts.

#### 46.9.3.12 Handling Errors in Alternative Command

In case of NACK generated by a slave device or SMBus timeout error, the TWI stops immediately the frame, but the DMA transfer may still be active. To prevent a new frame to be restarted with the remaining DMA data (transmit), the TWI prevents any start of frame until the FLEX_TWI_SR.LOCK flag is cleared.

The FLEX_TWI_SR.LOCK bit indicates the state of the TWI (locked or not locked).

When the TWI is locked, no transfer can begin until the LOCK is cleared using the FLEX_TWI_CR.LOCKCLR bit and until the error flags are cleared reading FLEX_TWI_SR.

In case of error, FLEX_TWI_THR may have been loaded with a new data. The FLEX_TWI_CR.THRCLR bit can be used to flush FLEX_TWI_THR. If the THRCLR bit is set, the TXRDY and TXCOMP flags are set.

#### 46.9.3.13 Read/Write Flowcharts

The flowcharts shown in this section provide examples for read and write operations. A polling or interrupt method can be used to check the status bits. The interrupt method requires that the Interrupt Enable Register (FLEX_TWI_IER) be configured first.

**Figure 46-95. TWI Write Operation with Single Data Byte without Internal Address**

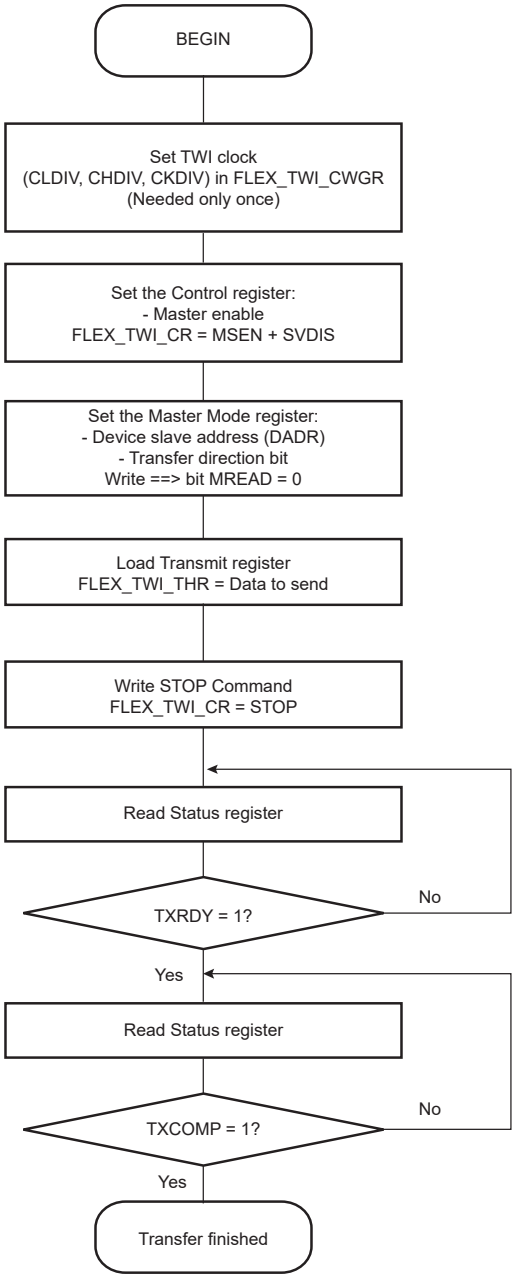


**Figure 46-96. TWI Write Operation with Single Data Byte and Internal Address**

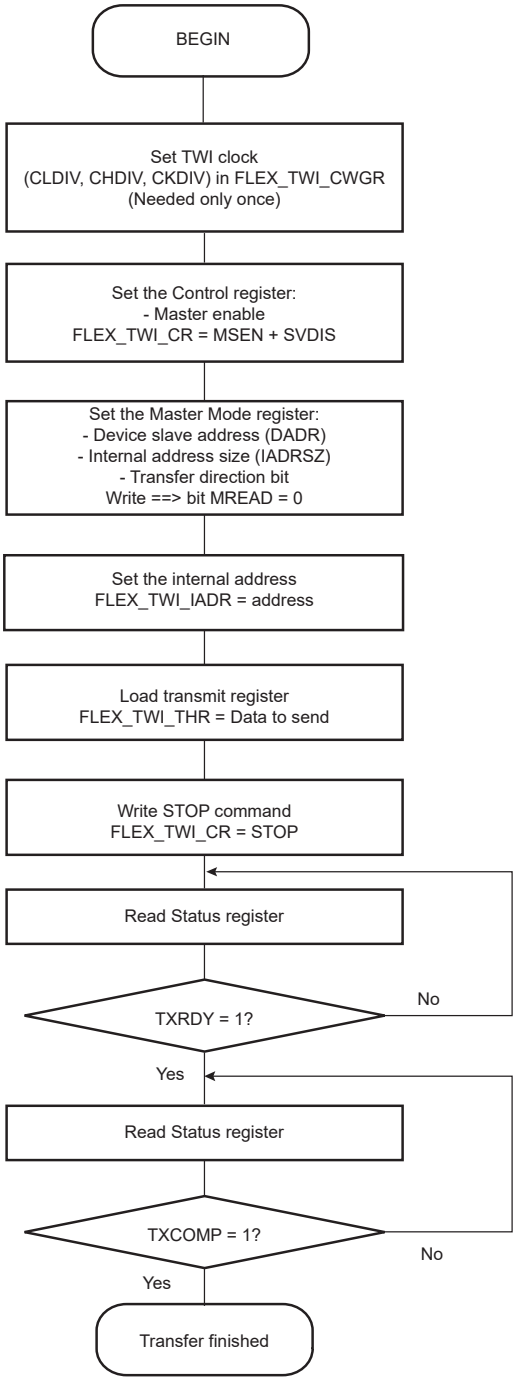


**Figure 46-97. TWI Write Operation with Multiple Data Bytes with or without Internal Address**

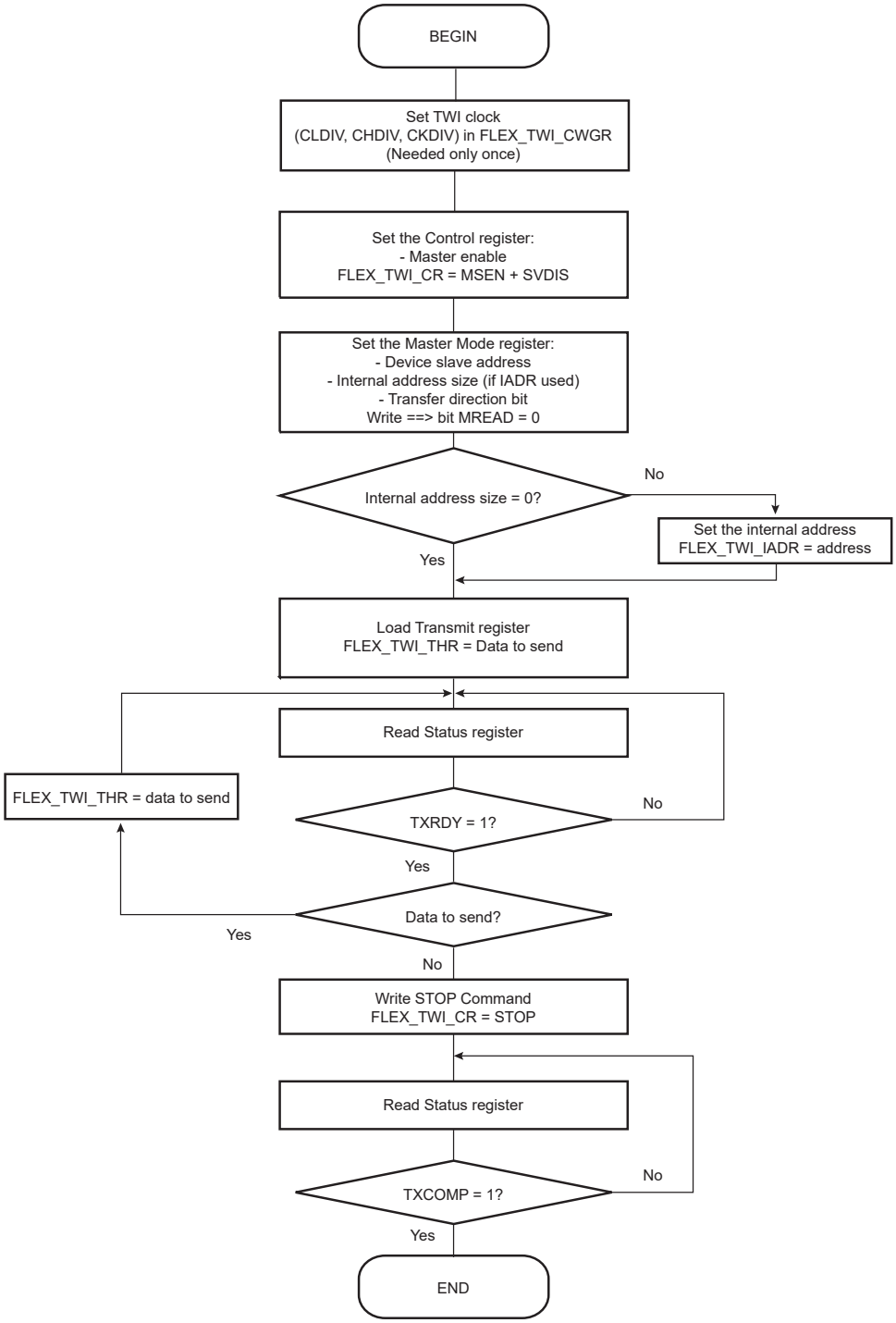


**Figure 46-98. SMBus Write Operation with Multiple Data Bytes with or without Internal Address and PEC Sending**

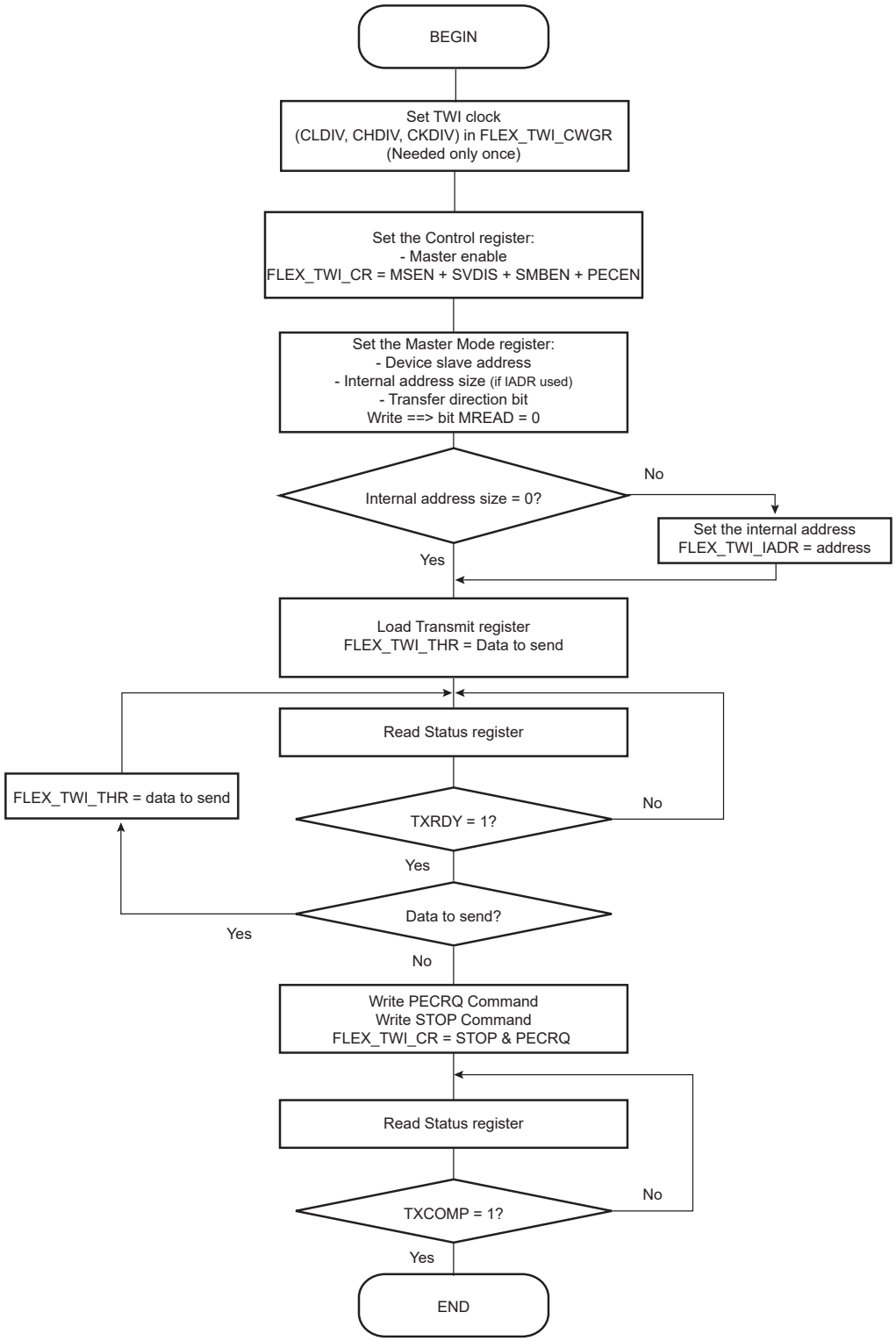


**Figure 46-99. SMBus Write Operation with Multiple Data Bytes with PEC and Alternative Command Mode**

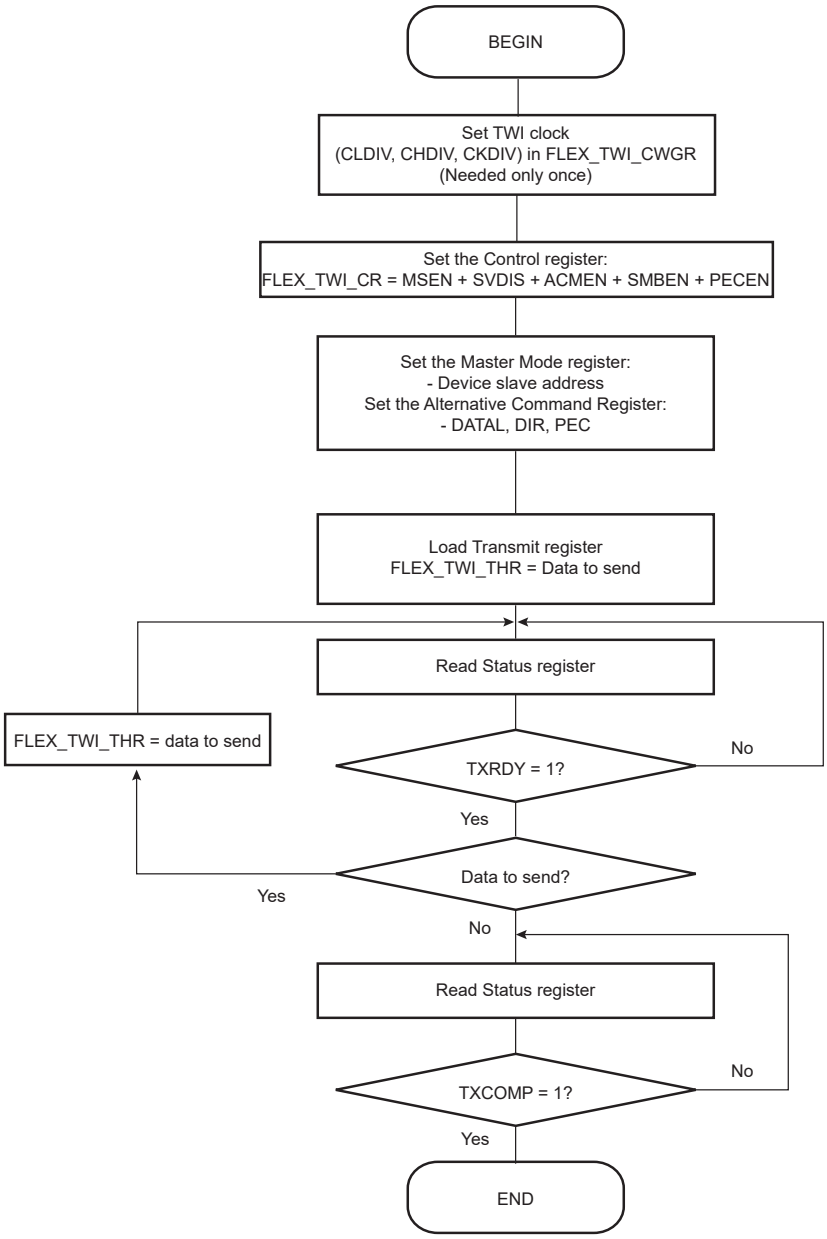


**Figure 46-100. TWI Write Operation with Multiple Data Bytes and Read Operation with Multiple Data Bytes (Sr)**

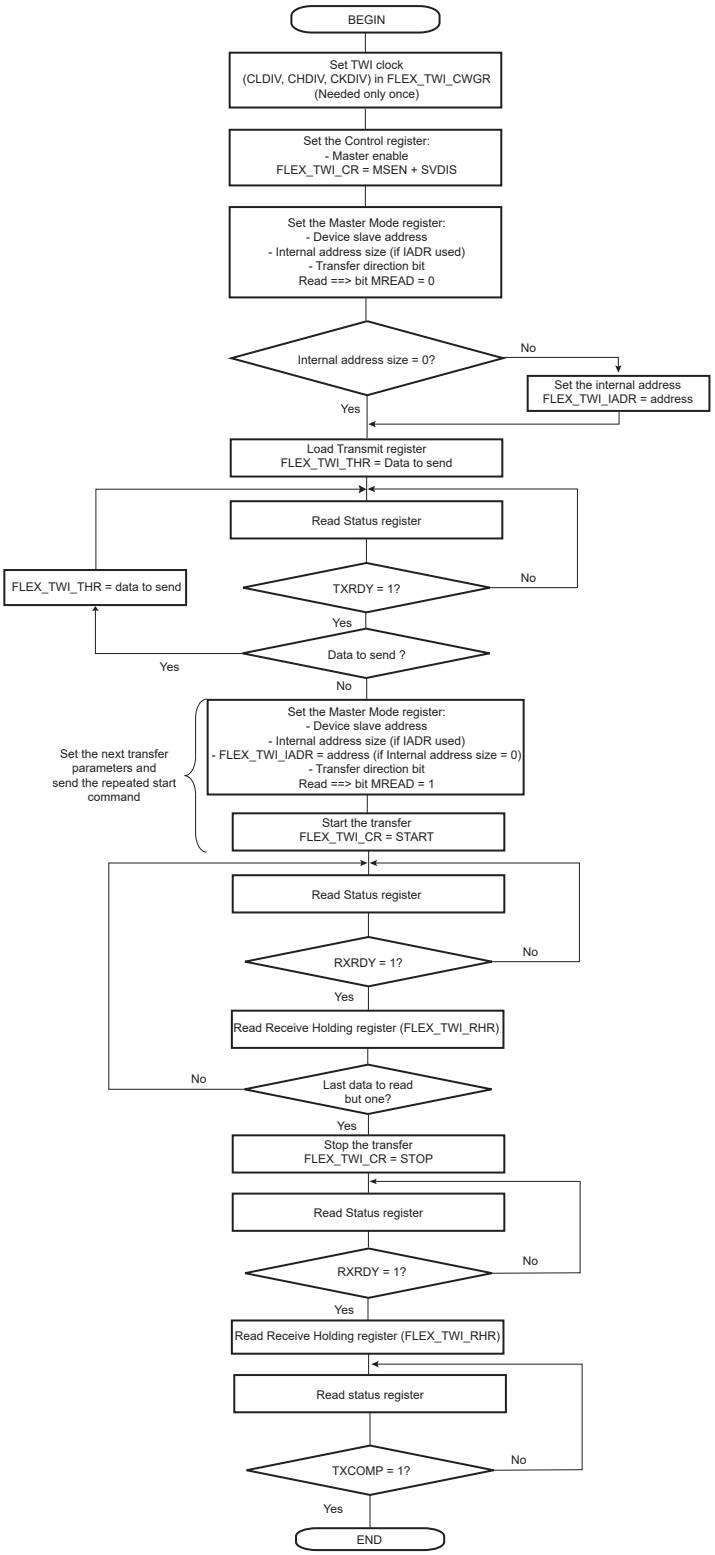


**Figure 46-101. TWI Write Operation with Multiple Data Bytes + Read Operation and Alternative Command Mode + PEC**

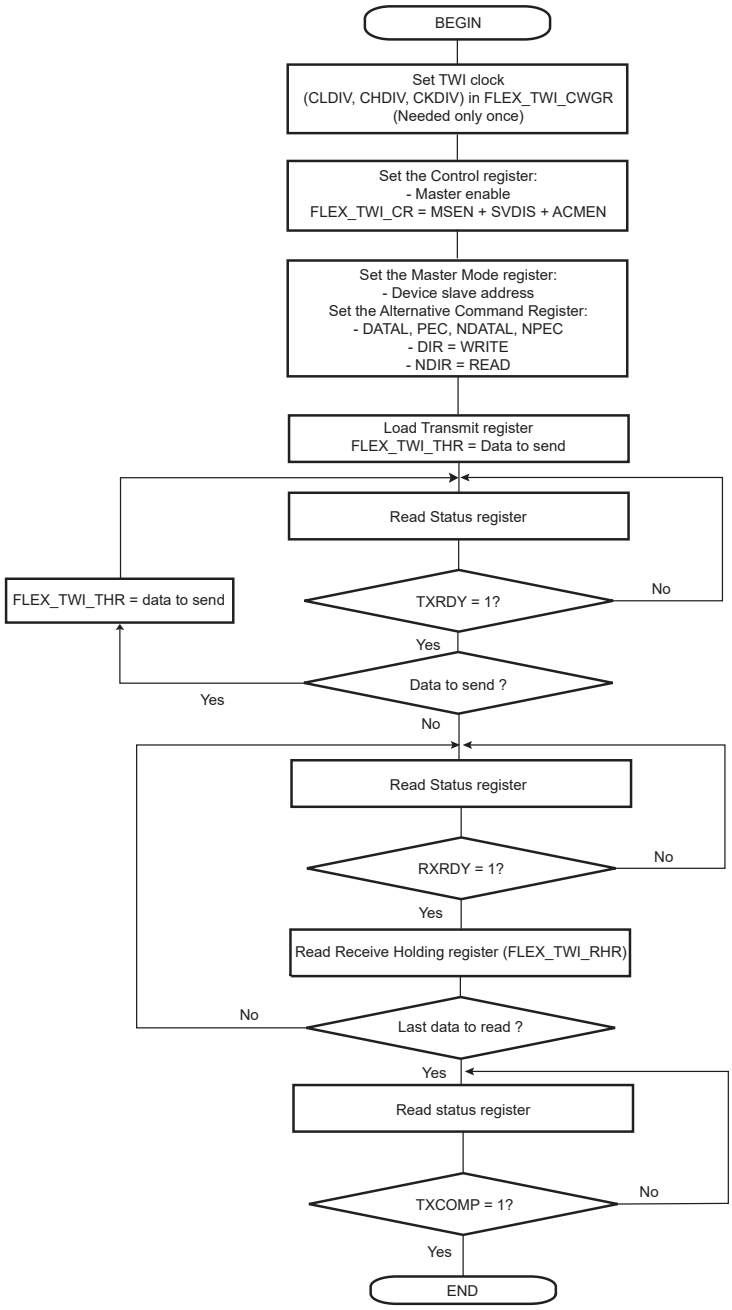


**Figure 46-102. TWI Read Operation with Single Data Byte without Internal Address**

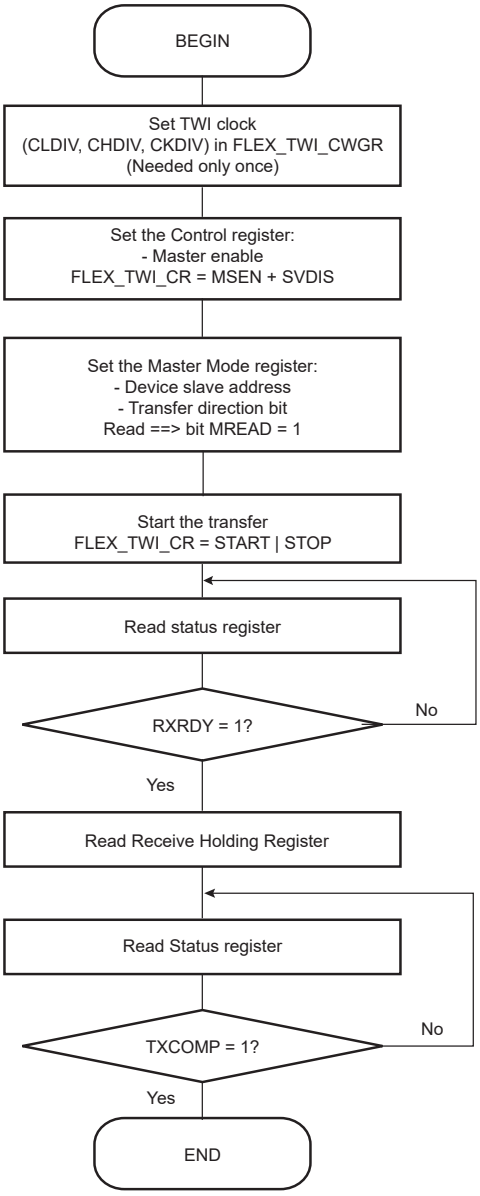


**Figure 46-103. TWI Read Operation with Single Data Byte and Internal Address**

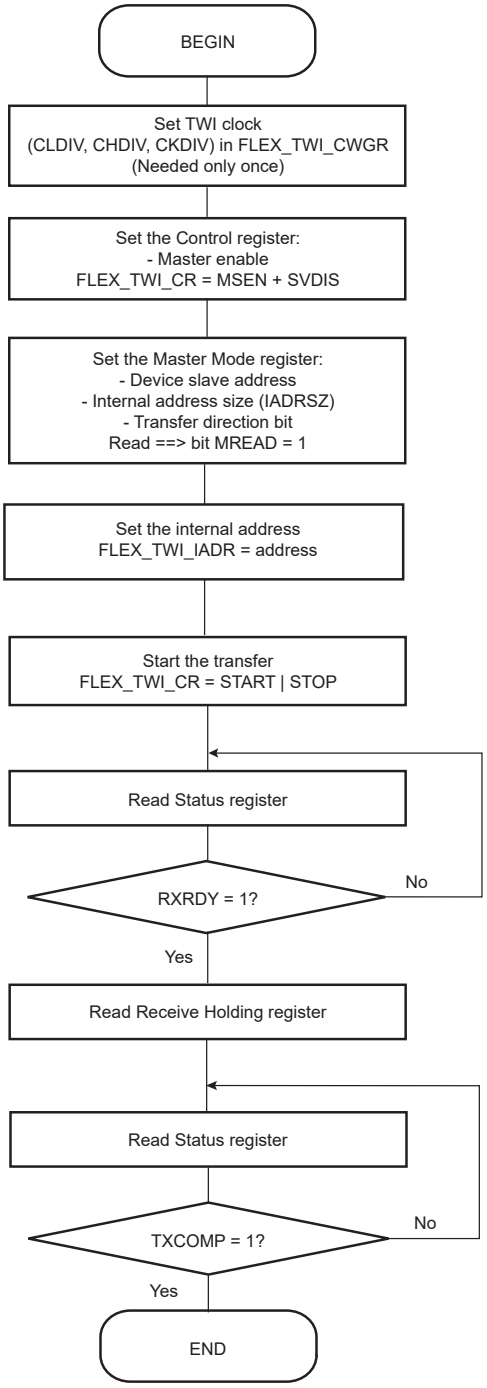


**Figure 46-104. TWI Read Operation with Multiple Data Bytes with or without Internal Address**

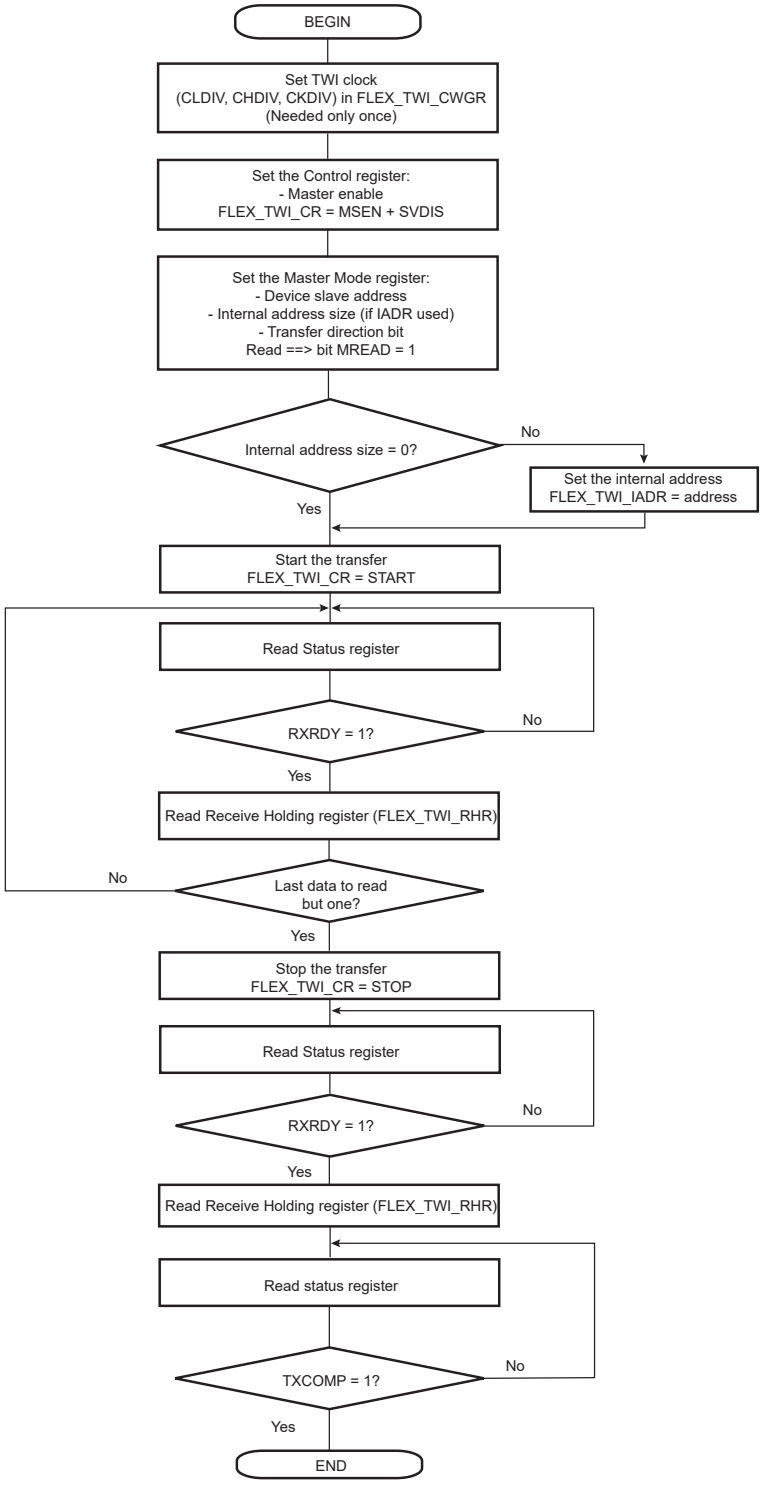


**Figure 46-105. TWI Read Operation with Multiple Data Bytes with or without Internal Address with PEC**

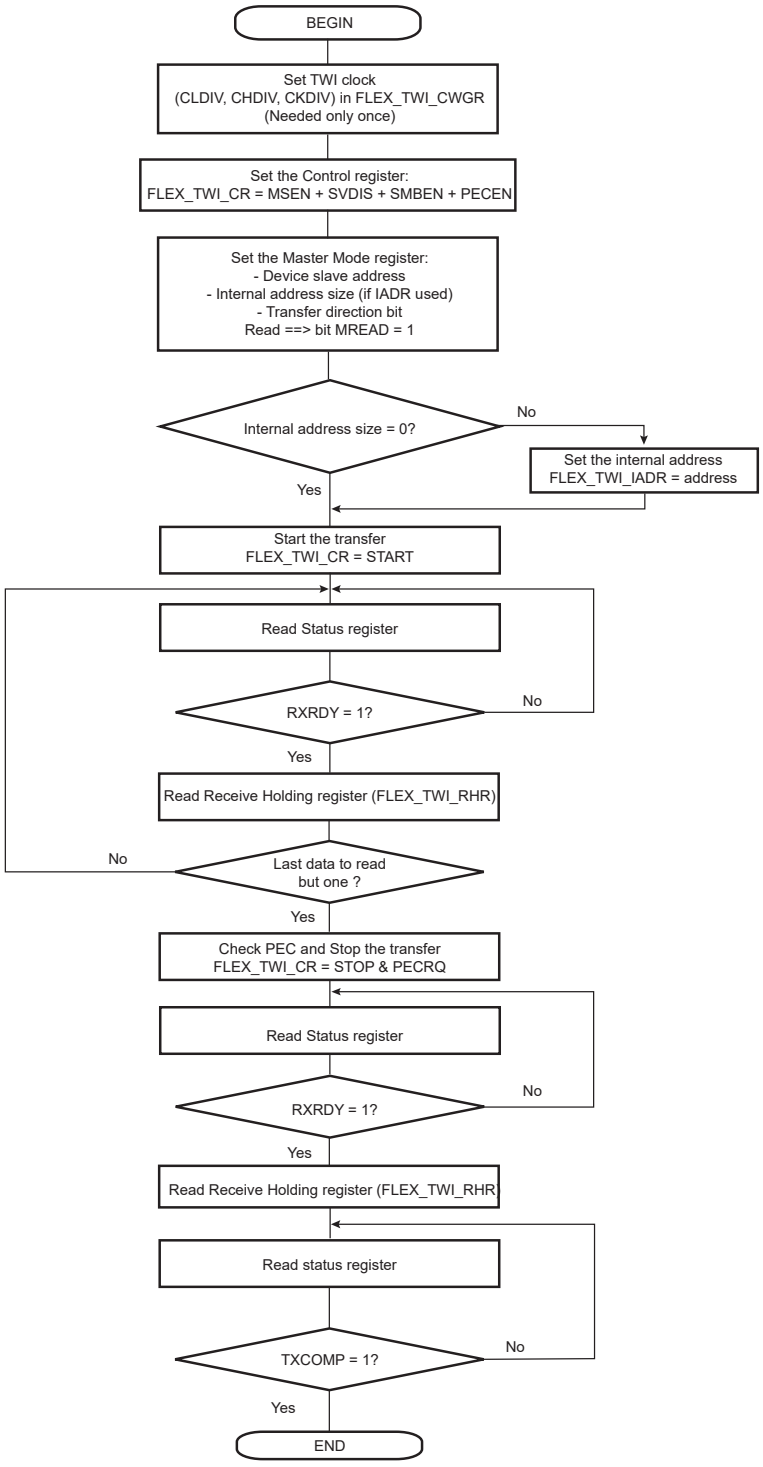


**Figure 46-106. TWI Read Operation with Multiple Data Bytes with Alternative Command Mode with PEC**

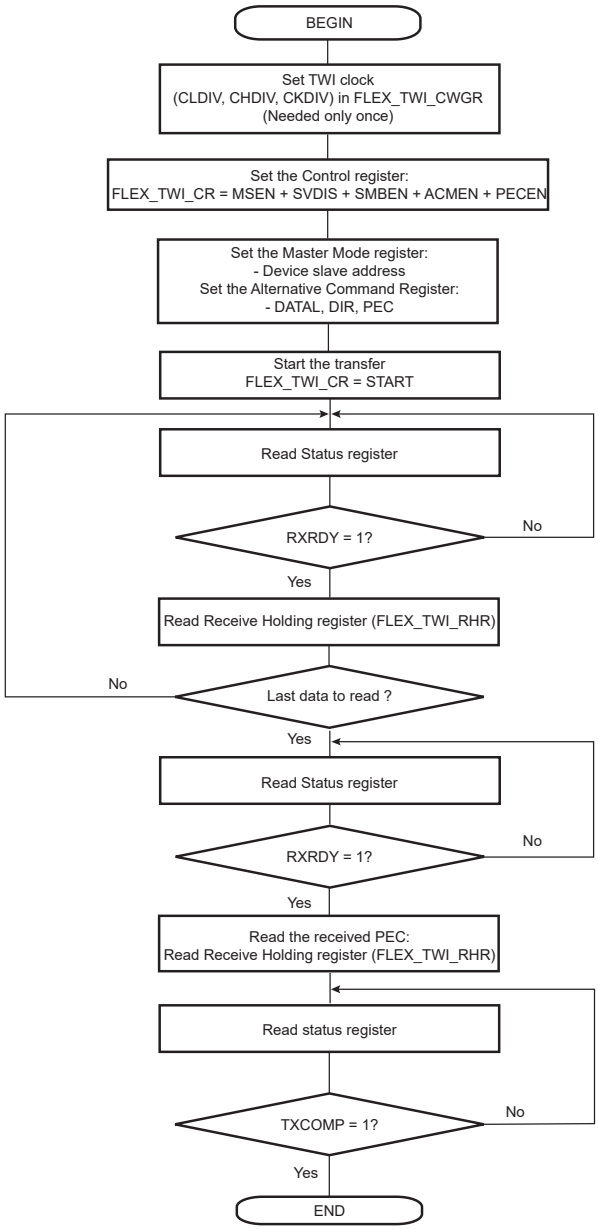


**Figure 46-107. TWI Read Operation with Multiple Data Bytes + Write Operation with Multiple Data Bytes (Sr)**

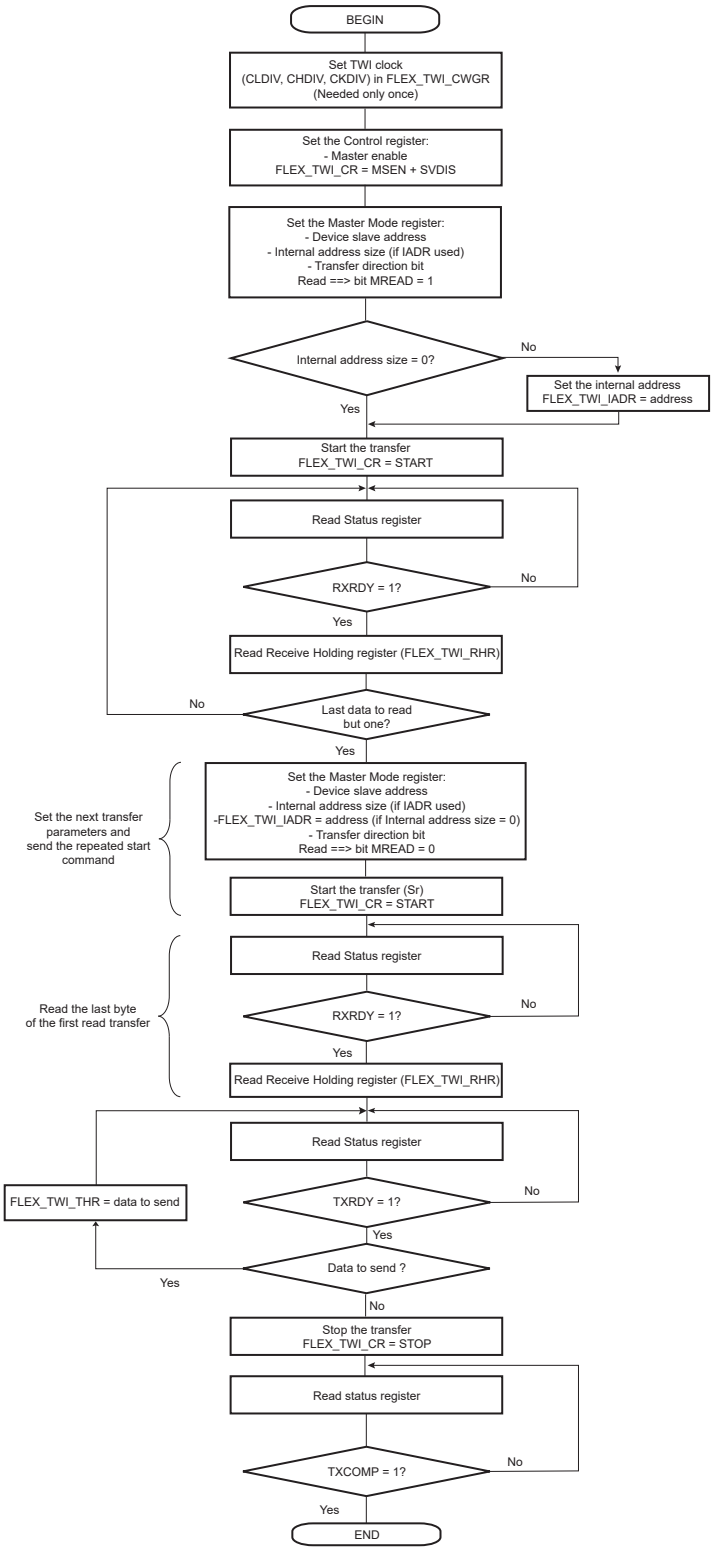
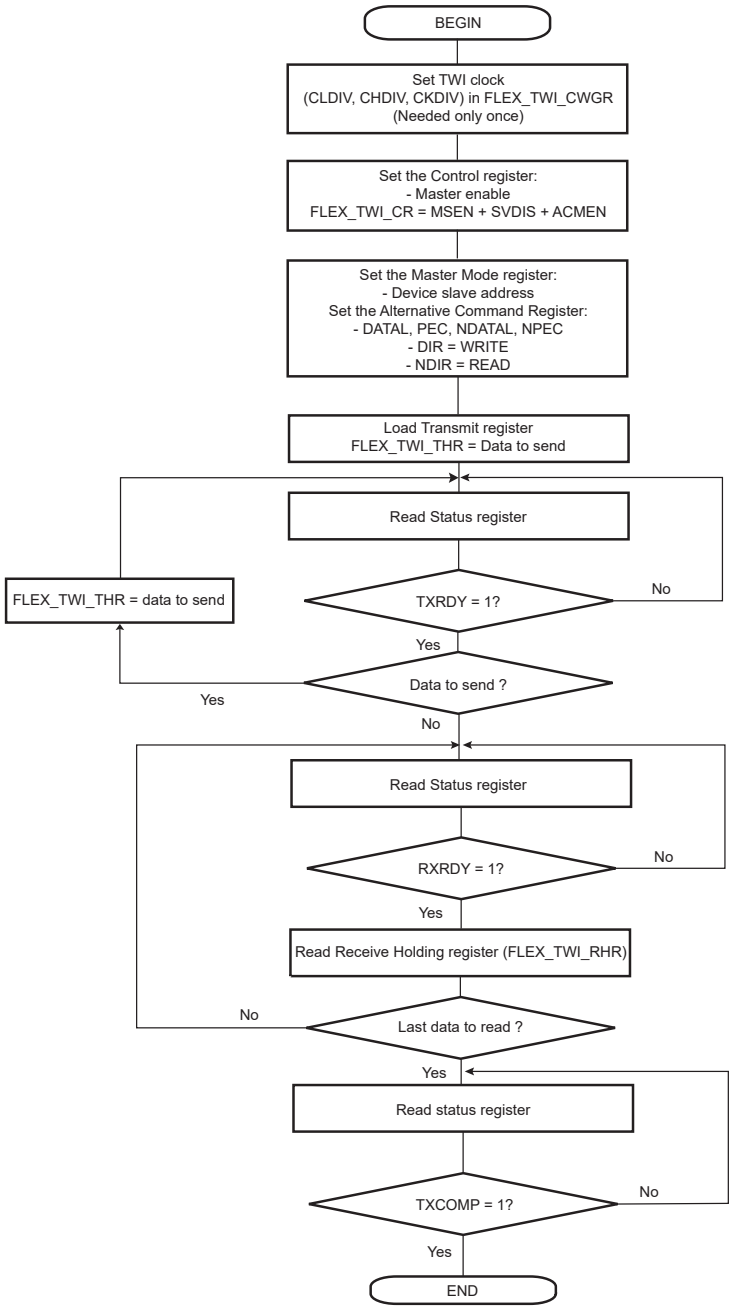


**Figure 46-108. TWI Read Operation with Multiple Data Bytes + Write with Alternative Command Mode with PEC**




### 46.9.4 Multi-Master Mode

#### 46.9.4.1 Definition

In Multi-Master mode, more than one master may handle the bus at the same time without data corruption by using arbitration.

Arbitration starts as soon as two or more masters place information on the bus at the same time, and stops (arbitration is lost) for the master that intends to send a logical one while the other master sends a logical zero.

As soon as arbitration is lost by a master, it stops sending data and listens to the bus in order to detect a STOP. When the STOP is detected, the master that has lost arbitration may put its data on the bus by respecting arbitration.

Arbitration is illustrated in figure "Arbitration Cases" below.

#### 46.9.4.2 Different Multi-Master Modes

Two Multi-Master modes may be distinguished:

- TWI as Master Only—TWI is considered as a master only and will never be addressed.
- TWI as Master or Slave—TWI may be either a master or a slave and may be addressed.

**Note:** Arbitration in supported in both Multi-Master modes.

##### 46.9.4.2.1 TWI as Master Only

In this mode, the TWI is considered as a master only (MSEN is always at one) and must be driven like a master with the ARBLST (ARBitration Lost) flag in addition.

If arbitration is lost (ARBLST = 1), the user must reinitiate the data transfer.

If the user starts a transfer (ex.: DADR + START + W + Write in THR) and if the bus is busy, the TWI automatically waits for a STOP condition on the bus to initiate the transfer (see figure "User Sends Data While the Bus is Busy" below).

**Note:** The state of the bus (busy or free) is not indicated in the user interface.

##### 46.9.4.2.2 TWI as Master or Slave

The automatic reversal from master to slave is not supported in case of a lost arbitration.

Then, in the case where TWI may be either a master or a slave, the user must manage the pseudo Multi-Master mode described in the steps below:

1. Program the TWI in Slave mode (SADR + MSDIS + SVEN) and perform a slave access (if TWI is addressed).
2. If the TWI has to be set in Master mode, wait until TXCOMP flag is at 1.
3. Program the Master mode (DADR + SVDIS + MSEN) and start the transfer (ex: START + Write in THR).
4. As soon as the Master mode is enabled, the TWI scans the bus in order to detect if it is busy or free. When the bus is considered as free, the TWI initiates the transfer.
5. As soon as the transfer is initiated and until a STOP condition is sent, the arbitration becomes relevant and the user must monitor the ARBLST flag.
6. If the arbitration is lost (ARBLST is = 1), the user must program the TWI in Slave mode in case the master that won the arbitration needs to access the TWI.
7. If the TWI has to be set in Slave mode, wait until TXCOMP flag is at 1 and then program the Slave mode.
   **Note:** In case the arbitration is lost and the TWI is addressed, the TWI will not acknowledge even if it is programmed in Slave mode as soon as ARBLST = 1. Then the master must repeat SADR.

**Figure 46-109. User Sends Data While the Bus is Busy**

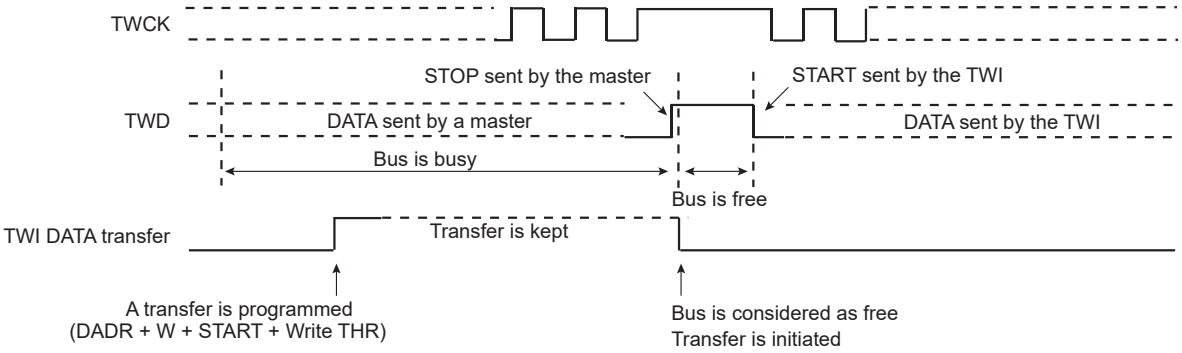
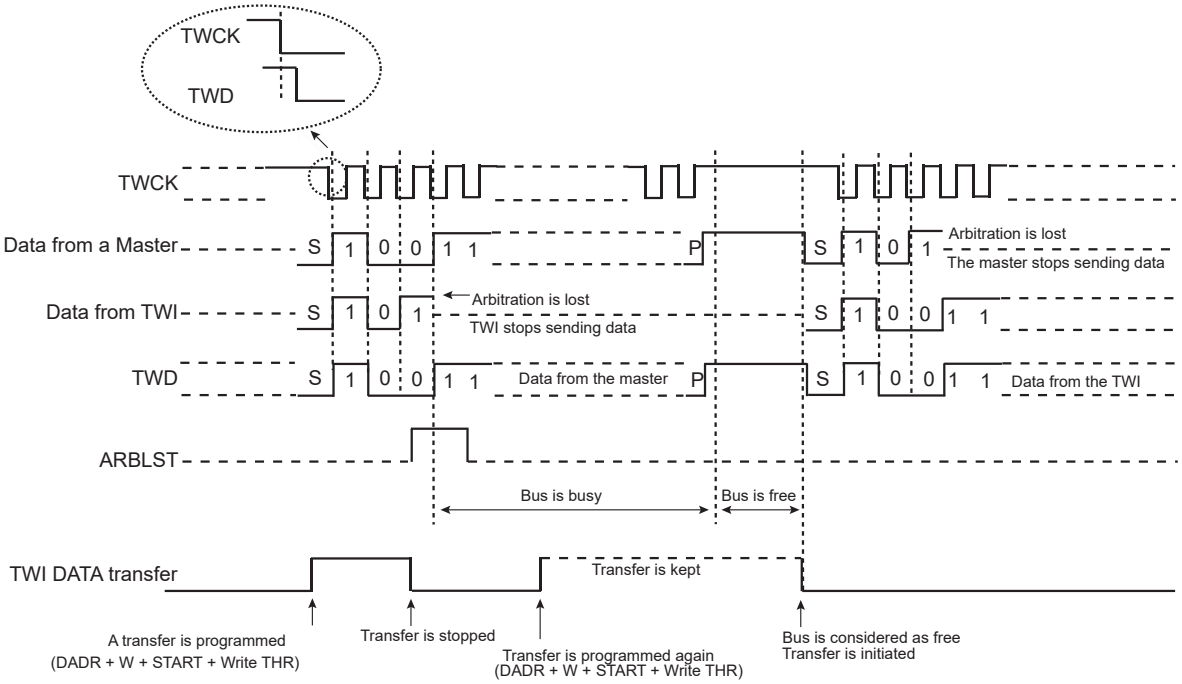
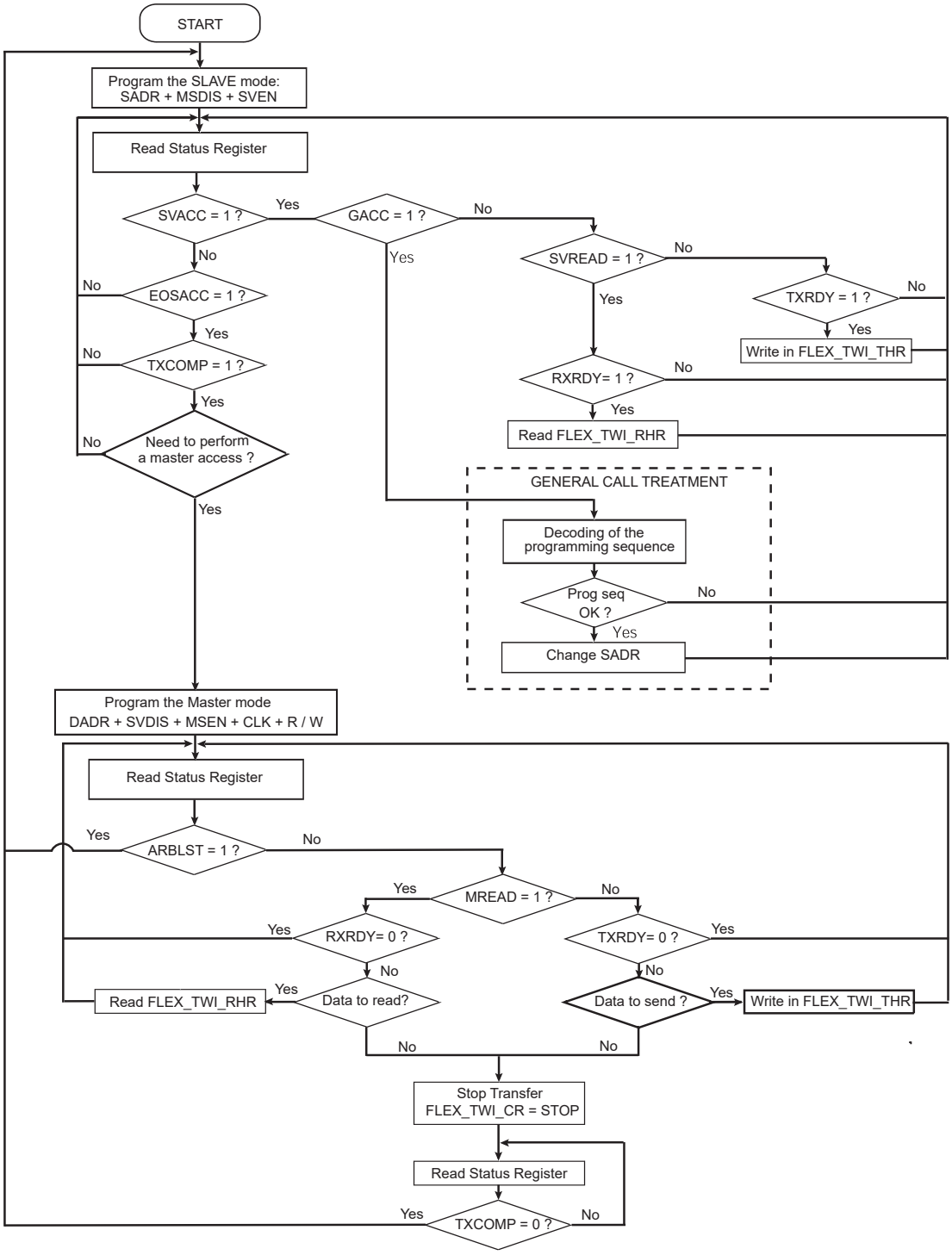



**Figure 46-110. Arbitration Cases**




The flowchart shown in the following figure gives an example of read and write operations in Multi-Master mode.

**Figure 46-111. Multi-Master Mode**




## 46.9.5 Slave Mode

### 46.9.5.1 Definition

Slave mode is defined as a mode where the device receives the clock and the address from another device called the master.

In this mode, the device never initiates and never completes the transmission (START, REPEATED_START and STOP conditions are always provided by the master).

#### 46.9.5.2 Programming Slave Mode

The following fields must be programmed before entering Slave mode:

1. FLEX_TWI_SMR.SADR: The slave device address is used in order to be accessed by master devices in Read or Write mode.
2. (Optional) FLEX_TWI_SMR.MASK can be set to mask some SADR address bits and thus allow multiple address matching.
3. FLEX_TWI_CR.MSDIS: Disables the Master mode.
4. FLEX_TWI_CR.SVEN: Enables the Slave mode.

As the device receives the clock, values written in FLEX_TWI_CWGR are not processed.

#### 46.9.5.3 Receiving Data

After a START or repeated START condition is detected, and if the address sent by the master matches the slave address programmed in the SADR (Slave Address) field, the SVACC (Slave Access) flag is set and SVREAD (Slave Read) indicates the direction of the transfer.

SVACC remains high until a STOP condition or a repeated START is detected. When such a condition is detected, EOSACC (End Of Slave Access) flag is set.

##### 46.9.5.3.1 Read Sequence

In the case of a read sequence (SVREAD is high), the TWI transfers data written in FLEX_TWI_THR (TWI Transmit Holding Register) until a STOP condition or a REPEATED_START + an address different from SADR is detected. Note that at the end of the read sequence TXCOMP (Transmission Complete) flag is set and SVACC is reset.

As soon as data is written in FLEX_TWI_THR, the TXRDY (Transmit Holding Register Ready) flag is reset, and it is set when the internal shifter is empty and the sent data acknowledged or not. If the data is not acknowledged, the NACK flag is set.

Note that a STOP or a repeated START always follows a NACK.

See figure "Read Access Ordered by a Master" below.

**Note:** To clear the TXRDY flag in Slave mode, write the FLEX_TWI_CR.SVDIS bit to 1, then write the FLEX_TWI_CR.SVEN bit to 1.

##### 46.9.5.3.2 Write Sequence

In the case of a write sequence (SVREAD is low), the RXRDY (Receive Holding Register Ready) flag is set as soon as a character has been received in FLEX_TWI_RHR (TWI Receive Holding Register). RXRDY is reset when reading FLEX_TWI_RHR.

TWI continues receiving data until a STOP condition or a REPEATED_START + an address different from SADR is detected. Note that at the end of the write sequence TXCOMP flag is set and SVACC reset.

See figure "Write Access Ordered by a Master" below.

##### 46.9.5.3.3 Clock Stretching Sequence

If FLEX_TWI_THR or FLEX_TWI_RHR is not written/read in time, the TWI performs a clock stretching.

Clock stretching information is given by the SCLWS (Clock Wait State) bit.

See figures "Clock Stretching in Read Mode" and "Clock Stretching in Write Mode" below.

**Note:** Clock stretching can be disabled by configuring the FLEX_TWI_SMR.SCLWSDIS bit. In that case, UNRE and OVRE flags will indicate underrun (when FLEX_TWI_THR is not filled on time) or overrun (when FLEX_TWI_RHR is not read on time).

#### 46.9.5.3.4 General Call

In the case where a GENERAL CALL is performed, the GACC (General Call Access) flag is set.

After GACC is set, it is up to the user to interpret the meaning of the GENERAL CALL and to decode the new address programming sequence.

See figure "Master Performs a General Call" below.

### 46.9.5.4 Data Transfer

#### 46.9.5.4.1 Read Operation

The Read mode is defined as a data requirement from the master.

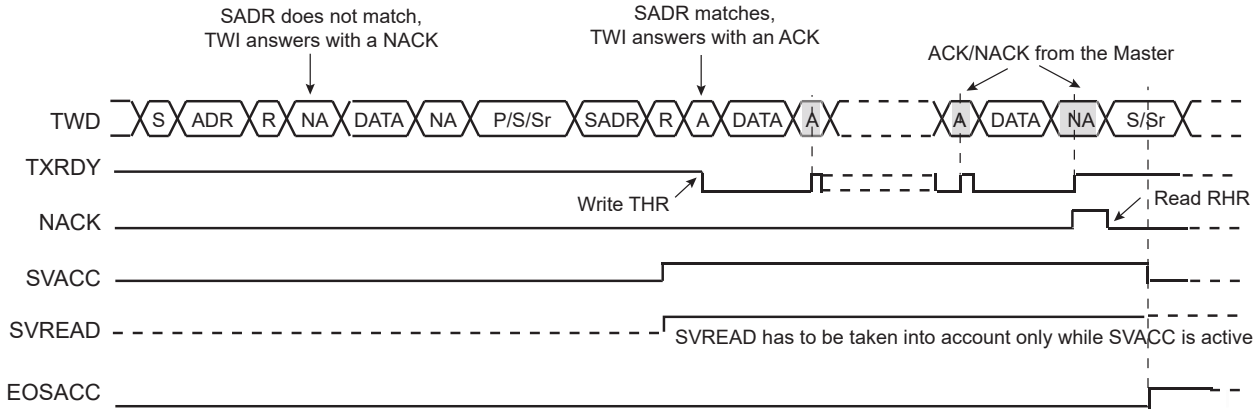
After a START or a REPEATED START condition is detected, the decoding of the address starts. If the slave address (SADR) is decoded, SVACC is set and SVREAD indicates the direction of the transfer.

Until a STOP or REPEATED START condition is detected, TWI continues sending data loaded in FLEX_TWI_THR.

If a STOP condition or a REPEATED START + an address different from SADR is detected, SVACC is reset.

The following figure describes the read operation.

**Figure 46-112. Read Access Ordered by a Master**




**Note:**

1. When SVACC is low, the state of SVREAD becomes irrelevant.
2. TXRDY is reset when data has been transmitted from FLEX_TWI_THR to the internal shifter and set when this data has been acknowledged or non acknowledged.

#### 46.9.5.4.2 Write Operation

The Write mode is defined as a data transmission from the master.

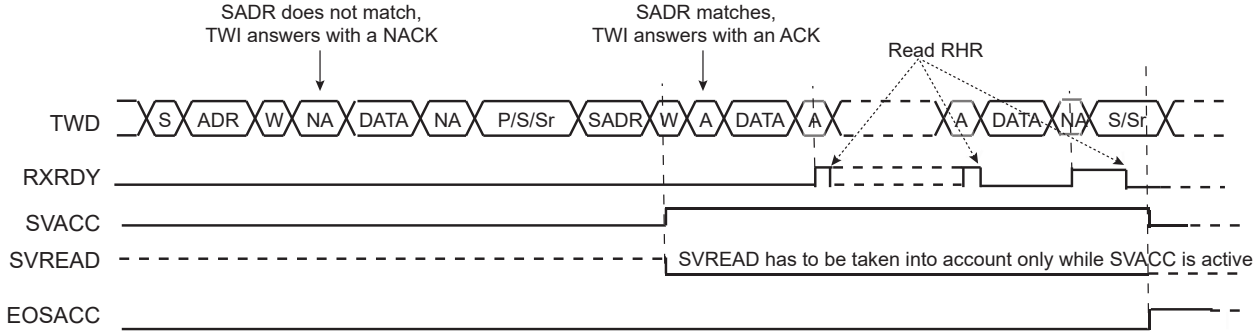
After a START or a REPEATED START, the decoding of the address starts. If the slave address is decoded, SVACC is set and SVREAD indicates the direction of the transfer (SVREAD is low in this case).

Until a STOP or REPEATED START condition is detected, TWI stores the received data in FLEX_TWI_RHR.

If a STOP condition or a REPEATED START + an address different from SADR is detected, SVACC is reset.

The following figure describes the write operation.

**Figure 46-113. Write Access Ordered by a Master**




**Note:**

1. When SVACC is low, the state of SVREAD becomes irrelevant.
2. RXRDY is set when data has been transmitted from the internal shifter to FLEX_TWI_RHR, and reset when this data is read.

#### 46.9.5.4.3 General Call

The general call is performed in order to change the address of the slave.

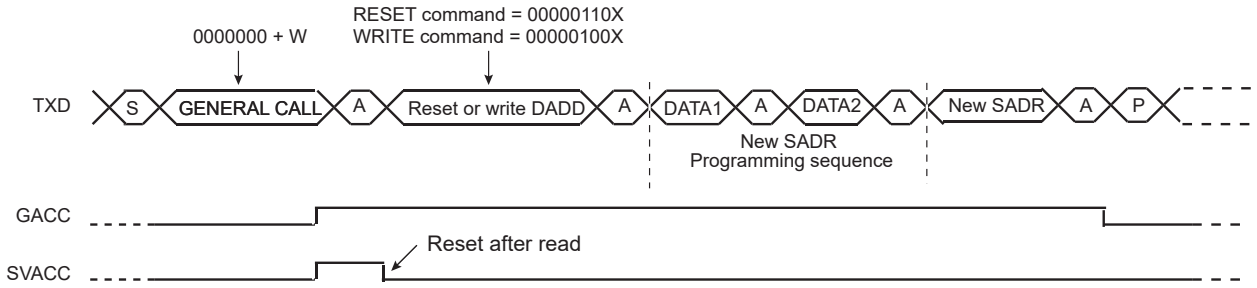
If a GENERAL CALL is detected, GACC is set.

After the detection of general call, it is up to the user to decode the commands which follow.

In case of a WRITE command, the user has to decode the programming sequence and program a new SADR if the programming sequence matches.

The following figure describes the general call access.

**Figure 46-114. Master Performs a General Call**




**Note:** This method enables to create a user-specific programming sequence by choosing the number of programming bytes. The programming sequence has to be provided to the master.

#### 46.9.5.4.4 Clock Stretching

In both Read and Write modes, it may happen that the FLEX_TWI_THR/FLEX_TWI_RHR buffer is not filled/emptied before the transmission/reception of a new character. In this case, to avoid sending/receiving undesired data, a clock stretching mechanism is implemented.

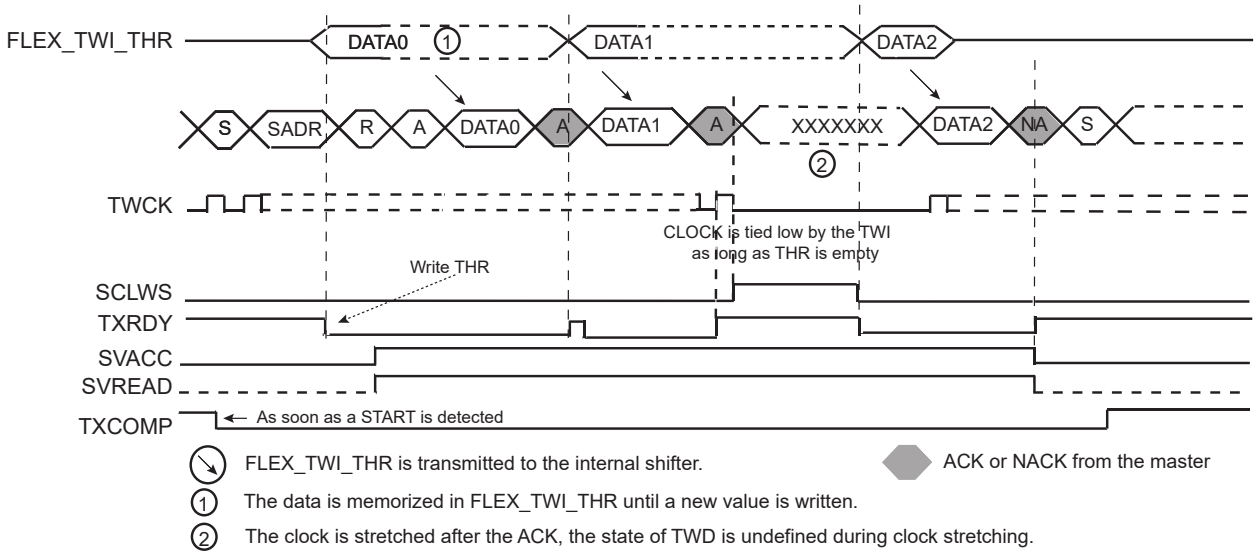
**Note:** Clock stretching can be disabled by setting the FLEX_TWI_SMR.SCLWSDIS bit. In that case, the UNRE and OVRE flags indicate an underrun (when FLEX_TWI_THR is not filled on time) or an overrun (when FLEX_TWI_RHR is not read on time).

***— Clock Stretching in Read Mode***

The clock is tied low if the internal shifter is empty and if a STOP or REPEATED START condition was not detected. It is tied low until the internal shifter is loaded.

The following figure describes clock stretching in Read mode.

**Figure 46-115. Clock Stretching in Read Mode**




**Note:**

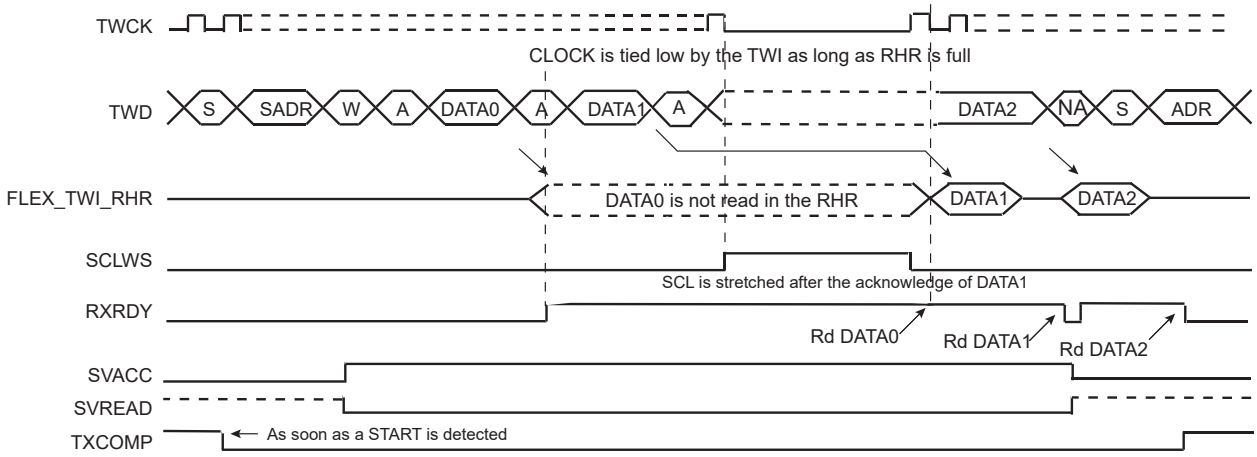
1. TXRDY is reset when data has been written in FLEX_TWI_THR to the internal shifter, and set when this data has been acknowledged or non acknowledged.
2. At the end of the read sequence, TXCOMP is set after a STOP or after a REPEATED_START + an address different from SADR.
3. SCLWS is automatically set when the clock stretching mechanism is started.

***— Clock Stretching in Write Mode***

The clock is tied low if the internal shifter and FLEX_TWI_RHR are full. If a STOP or REPEATED_START condition was not detected, it is tied low until FLEX_TWI_RHR is read.

The following figure describes the clock stretching in Write mode.

**Figure 46-116. Clock Stretching in Write Mode**




**Note:**

1. At the end of the read sequence, TXCOMP is set after a STOP or after a REPEATED_START + an address different from SADR.

2. SCLWS is automatically set when the clock stretching mechanism is started and automatically reset when the mechanism is finished.

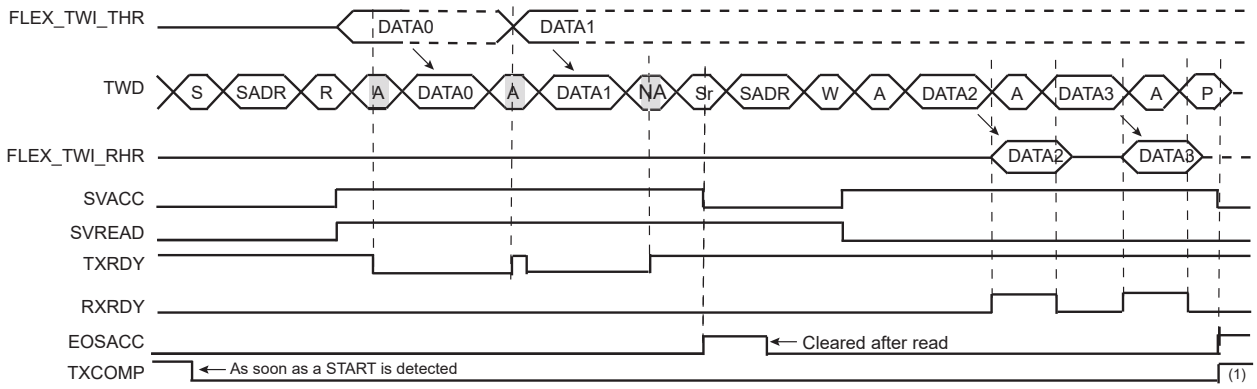
#### 46.9.5.4.5 Reversal after a Repeated Start

***— Reversal of Read to Write***

The master initiates the communication by a read command and finishes it by a write command.

The following figure describes the repeated start and the reversal from Read mode to Write mode.

**Figure 46-117. Repeated Start and Reversal from Read Mode to Write Mode**



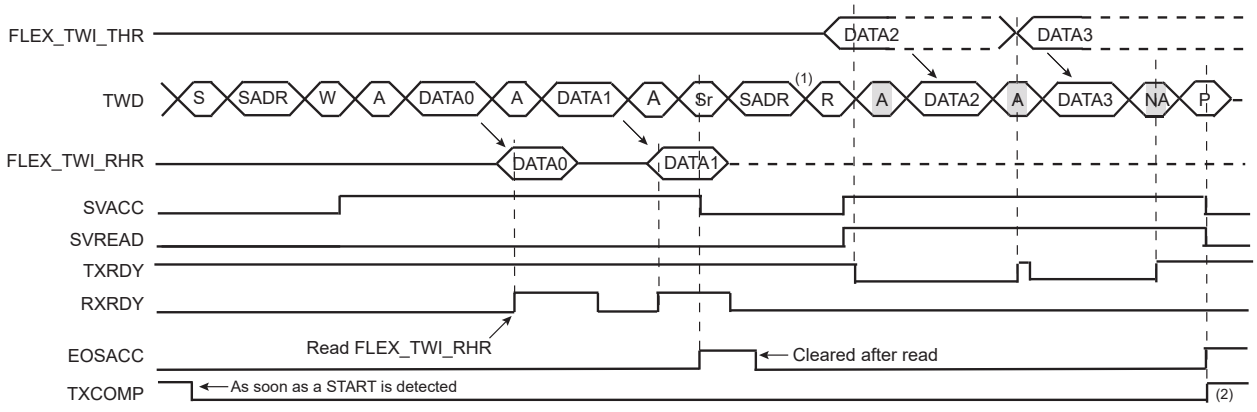

Note:
1. TXCOMP is only set at the end of the transmission because after the repeated start, SADR is detected again.

***— Reversal of Write to Read***

The master initiates the communication by a write command and finishes it by a read command.

The following figure describes the repeated start and the reversal from Write mode to Read mode.

**Figure 46-118. Repeated Start and Reversal from Write Mode to Read Mode**




Notes:
1. In this case, if FLEX_TWI_THR has not been written at the end of the read command, the clock is automatically stretched before the ACK.
2. TXCOMP is only set at the end of the transmission because after the repeated start, SADR is detected again.

#### 46.9.5.4.6 SMBus Mode

SMBus mode is enabled when the FLEX_TWI_CR.SMEN bit is written to one. SMBus mode operation is similar to I²C operation with the following exceptions:

1. Only 7-bit addressing can be used.
2. The SMBus standard describes a set of timeout values to ensure progress and throughput on the bus. These timeout values must be programmed into FLEX_TWI_SMBTR.
3. Transmissions can optionally include a CRC byte, called Packet Error Check (PEC).

4. A set of addresses have been reserved for protocol handling, such as alert response address (ARA) and host header (HH) address. Address matching on these addresses can be enabled by configuring FLEX_TWI_CR appropriately.

***— Packet Error Checking***

Each SMBus transfer can optionally end with a CRC byte, called the PEC byte. Writing the FLEX_TWI_CR.PECEN bit to one will send/check the FLEX_TWI_ACR.PEC field in the current transfer. The PEC generator is always updated on every bit transmitted or received, so that PEC handling on following linked transfers will be correct.

In Slave Receiver mode, the master calculates a PEC value and transmits it to the slave after all data bytes have been transmitted. Upon reception of this PEC byte, the slave will compare it to the PEC value it has computed itself. If the values match, the data was received correctly, and the slave will return an ACK to the master. If the PEC values differ, data was corrupted, and the slave will return a NACK value. The FLEX_TWI_SR.PECERR bit is set automatically if a PEC error occurred.

In Slave Transmitter mode, the slave calculates a PEC value and transmits it to the master after all data bytes have been transmitted. Upon reception of this PEC byte, the master will compare it to the PEC value it has computed itself. If the values match, the data was received correctly. If the PEC values differ, data was corrupted, and the master must take appropriate action.

See section Slave Read/Write Flowcharts for detailed flowcharts.

***— Timeouts***

The TWI SMBus Timing Register (FLEX_TWI_SMBTR) configures the SMBus timeout values. If a timeout occurs, the slave will leave the bus. Furthermore, the FLEX_TWI_SR.TOUT bit is set.

#### 46.9.5.5 High-Speed Slave Mode

High-speed mode is enabled when the FLEX_TWI_CR.HSEN bit is written to one. Furthermore, the analog pad filter must be enabled, the FLEX_TWI_FILTR.PADFEN bit must be written to one and the FLEX_TWI_FILTR.FILT bit must be cleared. TWI High-speed mode operation is similar to TWI operation with the following exceptions:

1. A master code is received first at normal speed before entering High-speed mode period.
2. When TWI High-speed mode is active, clock stretching is only allowed after acknowledge (ACK), not-acknowledge (NACK), START (S) or repeated START (Sr) (asa consequence, OVF may happen).

TWI High-speed mode allows transfers of up to 3.4 Mbit/s.

The TWI slave in High-speed mode requires that the peripheral clock runs at a minimum of 14 MHz if slave clock stretching is enabled (SCLWSDIS bit at ‘0’). If slave clock stretching is disabled (SCLWSDIS bit at ‘1’), the peripheral clock must run at a minimum of 11 MHz (assuming the system has no latency).

**Note:**

1. When slave clock stretching is disabled, FLEX_TWI_RHR must always be read before receiving the next data (MASTER write frame). It is strongly recommended to use either the polling method on the FLEX_TWI_SR.RXRDY flag, or the DMA. If the receive is managed by an interrupt, the TWI interrupt priority must be set to the right level and its latency minimized to avoid receive overrun.
2. When slave clock stretching is disabled, FLEX_TWI_THR must be filled with the first data to send before the beginning of the frame (MASTER read frame). It is strongly recommended to use either the polling method on the FLEX_TWI_SR.TXRDY flag, or the DMA. If the transmit is managed by an interrupt, the TWI interrupt priority must be set to the right level and its latency minimized to avoid transmit underrun.

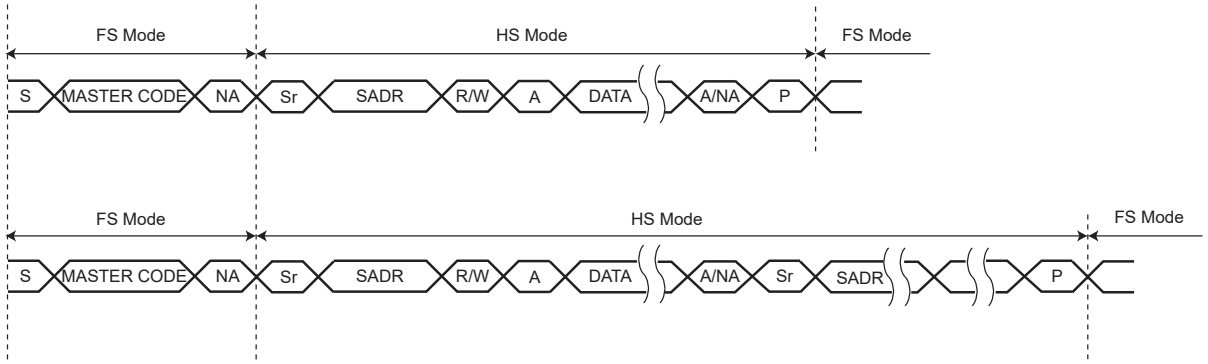
#### 46.9.5.5.1 Read/Write Operation

A TWI high-speed frame always begins with the following sequence:

1. START condition (S)
2. Master Code (0000 1XXX)
3. Not-acknowledge (NACK)

When the TWI is programmed in Slave mode and TWI High-speed mode is activated, master code matching is activated and internal timings are set to match the TWI High-speed mode requirements.

**Figure 46-119. High-Speed Mode Read/Write**




#### 46.9.5.5.2 Usage

TWI High-speed mode usage is the same as the standard TWI (see section Read/Write Flowcharts).

### 46.9.5.6 Alternative Command

In Slave mode, the Alternative Command mode is used when the SMBus mode is enabled to send or check the PEC byte.

The Alternative Command mode is enabled by setting the ACMEN bit of the TWIHS Control Register, and the transfer is configured in TWIHS_ACR.

For a combined transfer with PEC, only the NPEC bit in TWIHS_ACR must be set as the PEC byte is sent once at the end of the frame.

See section Slave Read/Write Flowcharts for detailed flowcharts.

### 46.9.5.7 TWI Asynchronous and Partial Wakeup (SleepWalking)

The TWI module includes an asynchronous start condition detector, it is capable of waking the device up from a Sleep mode upon an address match (and optionally an additional data match), including Sleep modes where the TWI peripheral clock is stopped.

The FLEX_TWI_RHR register must be read prior to enable the asynchronous and partial wakeup.

After detecting the START condition on the bus, the TWI will stretch TWCK until the TWI peripheral clock has started. The time required for starting the TWI peripheral depends on which Sleep mode the device is in. After the TWI peripheral clock has started, the TWI releases its TWCK stretching and receives one byte of data (slave address) on the bus. At this time, only a limited part of the device, including the TWI module, receives a clock, thus saving power. If the address phase causes a TWIS address match (and optionally if the first data byte causes data match as well), the entire device is wakened and normal TWI address matching actions are performed. Normal TWI transfer then follows. If the TWI module is not addressed (or if the optional data match fails), the TWI peripheral clock is automatically stopped and the device returns to its original Sleep mode.

The TWI module has the capability to match on more than one address. The FLEX_TWI_SMR.SADR1EN/SADR2EN/SADR3EN bits enable address matching on additional addresses which can be configured through the FLEX_TWI_SWMR.SADR1/SADR2/SADR3 fields. The SleepWalking matching process can be extended to the first received data byte if the FLEX_TWI_SMR.DATAMEN bit is set. In that case, a complete matching includes address matching and first received data matching. The FLEX_TWI_SWMR.DATAM field can be used to configure the data to match on the first received byte.

When the system is in Active mode and the TWI enters asynchronous partial Wakeup mode, the flag SVACC must be programmed as the unique source of the TWI interrupt and the data match comparison must be disabled.

When the system exits Wait mode as the result of a matching condition, the SVACC flag is used to determine if the TWI is the source of the exit from Wait mode.

**Figure 46-120. Address Match and Data Matching Disabled**

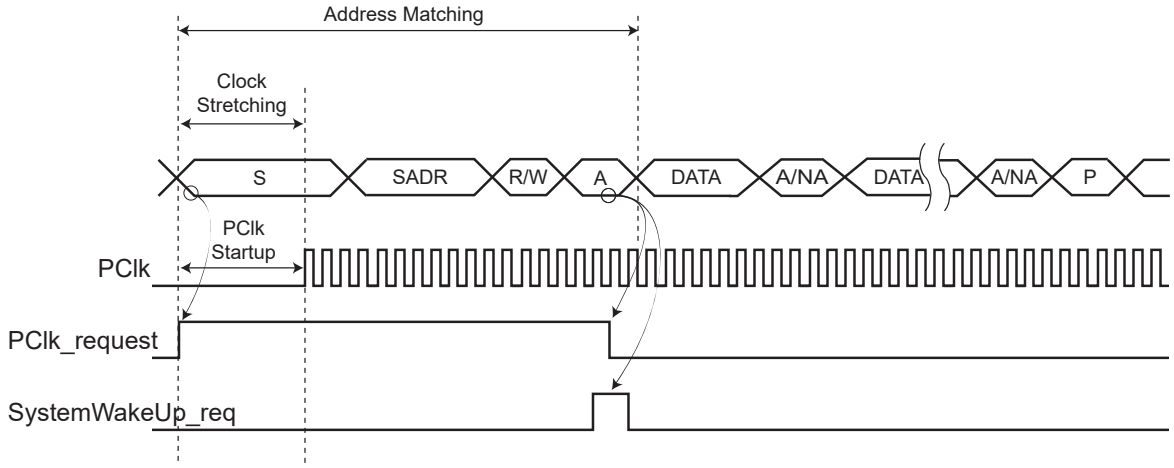



**Figure 46-121. Address Does Not Match and Data Matching Disabled**

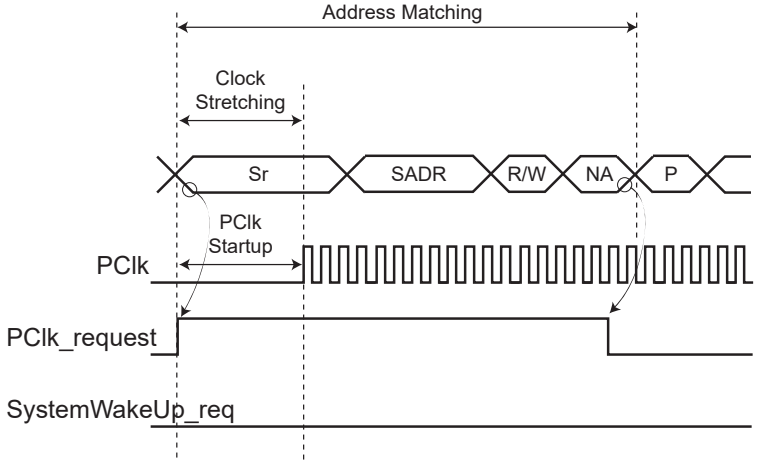


**Figure 46-122. Address and Data Match (Data Matching Enabled)**

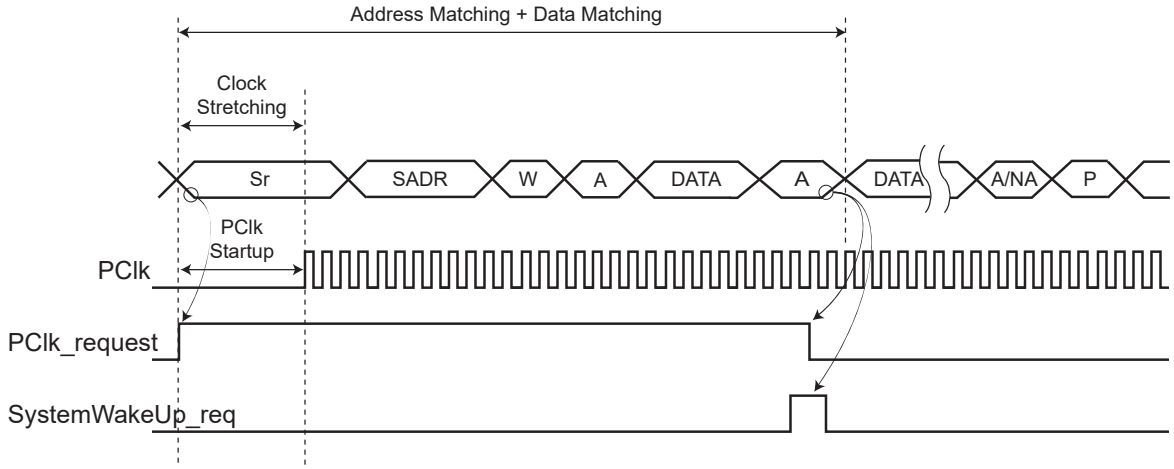
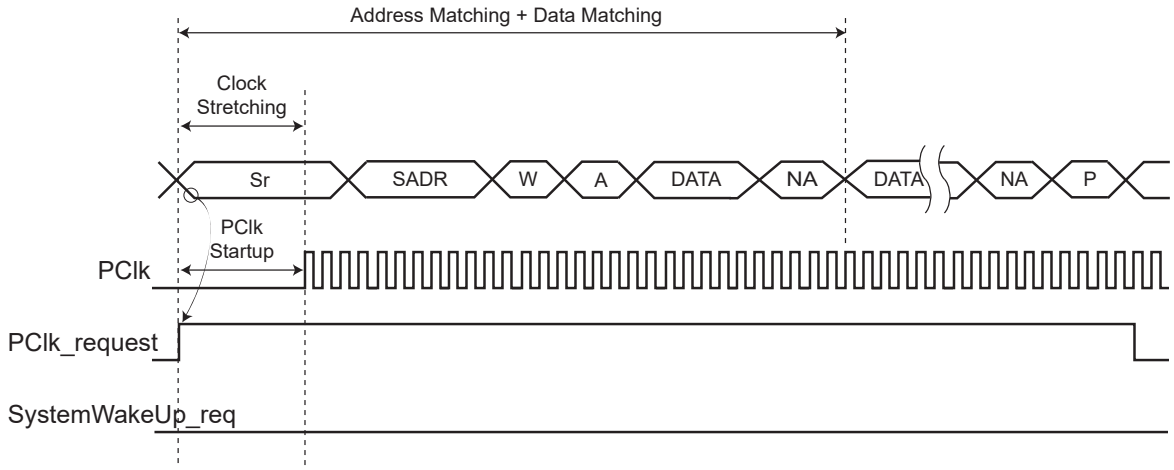



**Figure 46-123. Address Matches and Data Do Not Match (Data Matching Enabled)**




#### 46.9.5.8 Slave Read/Write Flowcharts

The flowchart shown in the following figure gives an example of read and write operations in Slave mode. A polling or interrupt method can be used to check the status bits. The interrupt method requires that the Interrupt Enable Register (FLEX_TWI_IER) be configured first.

**Figure 46-124. Read/Write in Slave Mode**

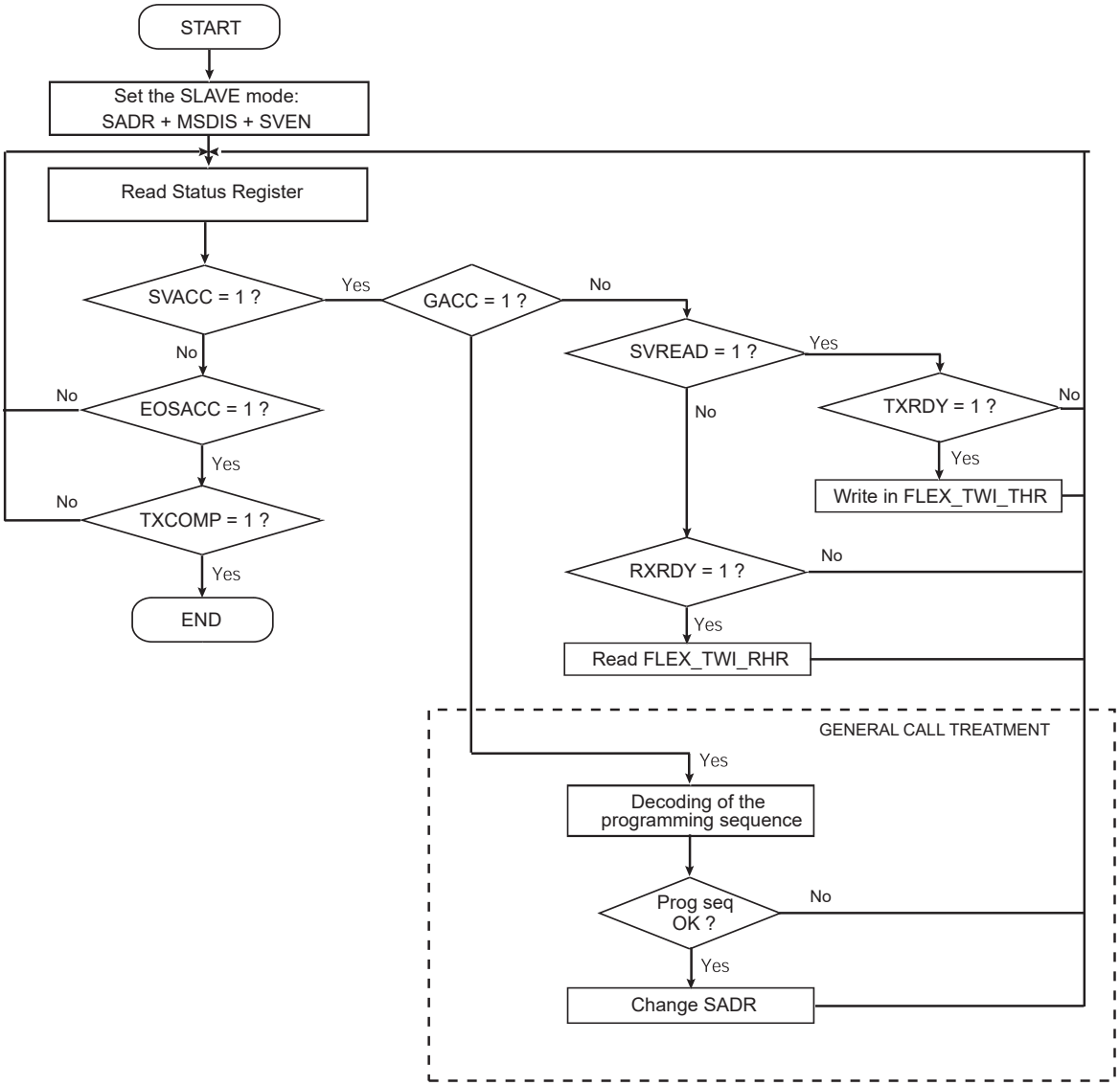


**Figure 46-125. Read/Write in Slave Mode with SMBus PEC**

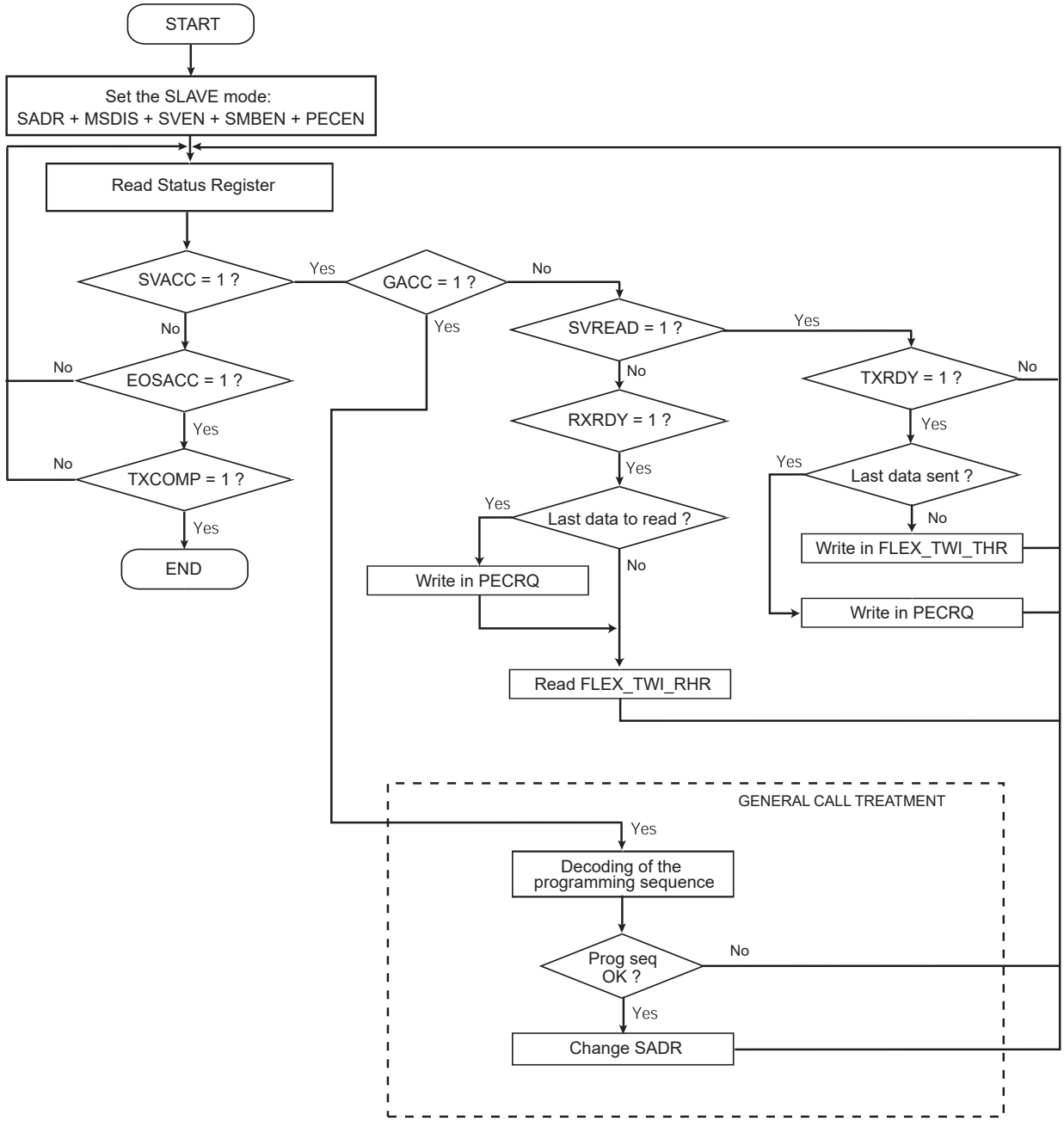
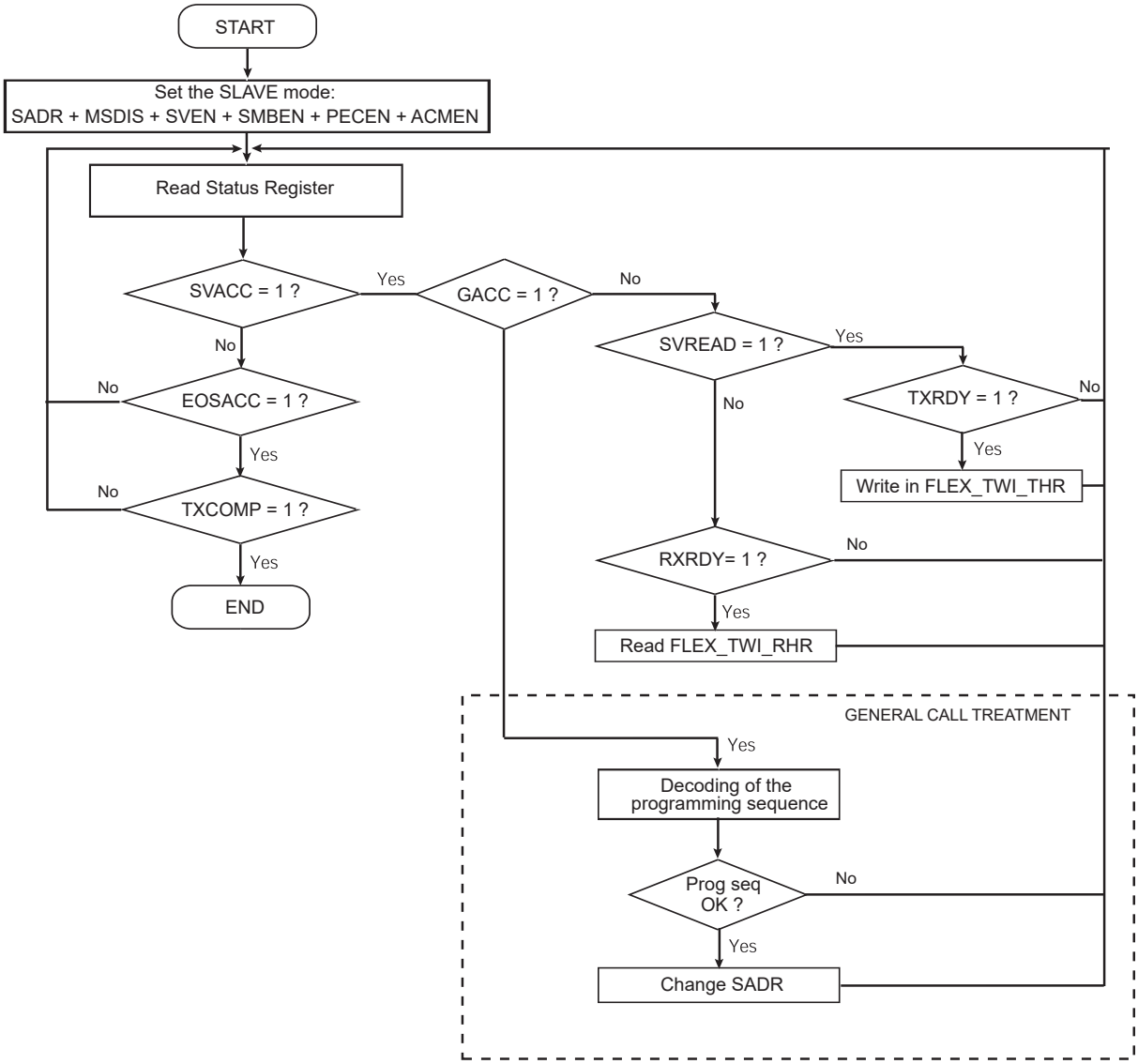


**Figure 46-126. Read/Write in Slave Mode with SMBus PEC and Alternative Command Mode**




### 46.9.6 TWI FIFOs

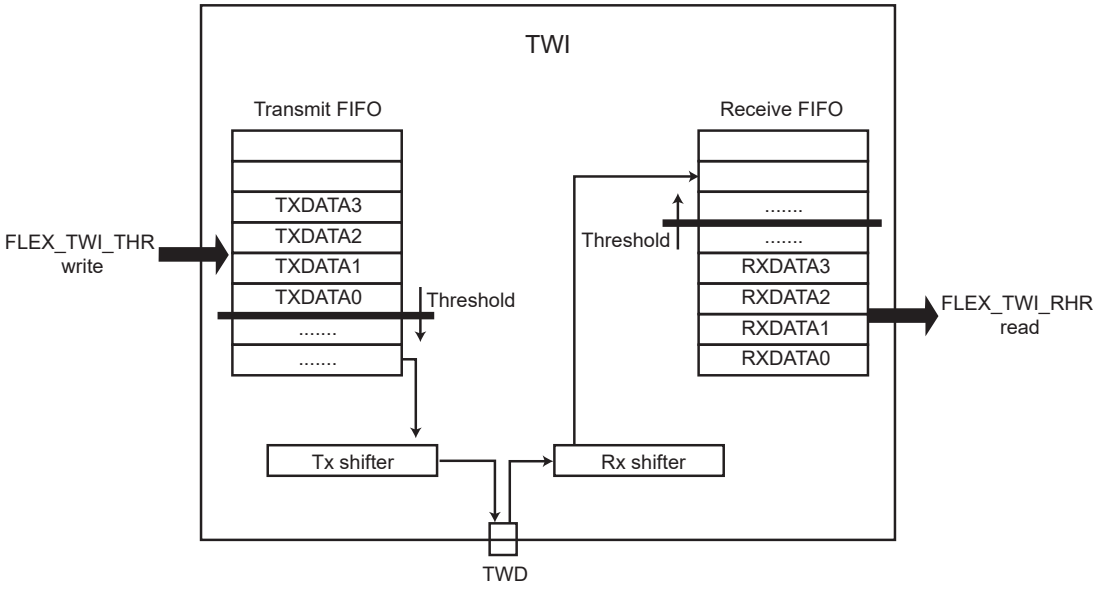
#### 46.9.6.1 Overview

The TWI includes two FIFOs which can be enabled/disabled using FLEX_TWI_CR.FIFOEN/FIFODIS. Both Master and Slave modes must be disabled before enabling or disabling the FIFOs (FLEX_TWI_CR.MSDIS/SVDIS).

Writing FLEX_TWI_CR.FIFOEN to ‘1’ enables a 16-data Transmit FIFO and a 16-data Receive FIFO.

It is possible to write or to read single or multiple data in the same access to FLEX_TWI_THR/RHR, depending on FLEX_TWI_FMR.TXRDYM/RXRDYM settings.

**Figure 46-127. FIFOs Block Diagram**




### 46.9.6.2 Sending Data with FIFO Enabled

When the Transmit FIFO is enabled, write access to FLEX_TWI_THR loads the Transmit FIFO.

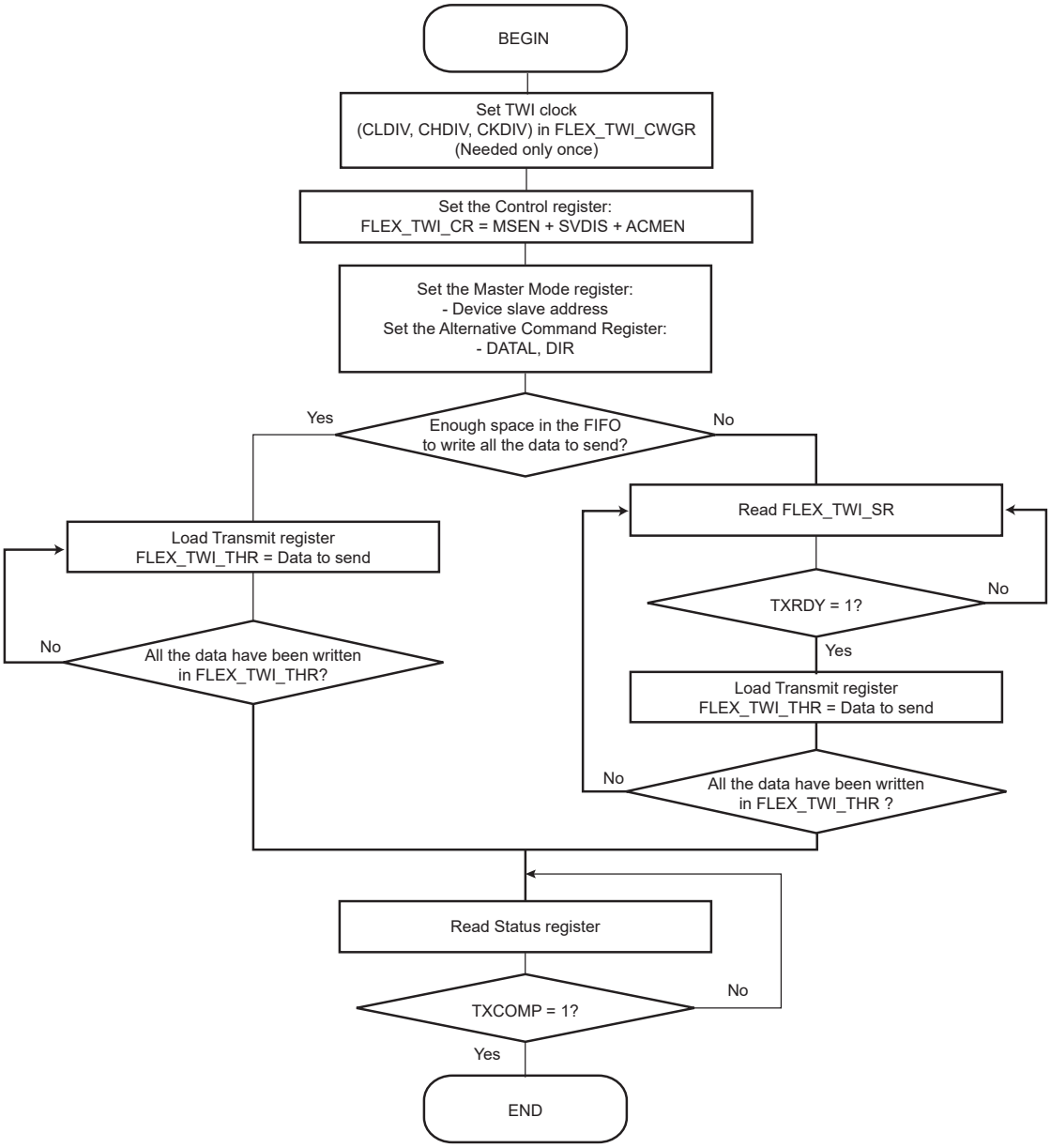
The Transmit FIFO level is provided in FLEX_TWI_FLR.TXFL. If the FIFO can accept the number of data to be transmitted, there is no need to monitor FLEX_TWI_SR.TXRDY and the data can be successively written in FLEX_TWI_THR.

If the FIFO cannot accept the data due to insufficient space, wait for the TXRDY flag to be set before writing the data in FLEX_TWI_THR.

When the space in the FIFO allows only a portion of the data to be written, the TXRDY flag must be monitored before writing the remaining data.

See figures Sending Data with FIFO Enabled in Master Mode and Sending/Receiving Data with FIFO Enabled in Slave Mode.

**Figure 46-128. Sending Data with FIFO Enabled in Master Mode**



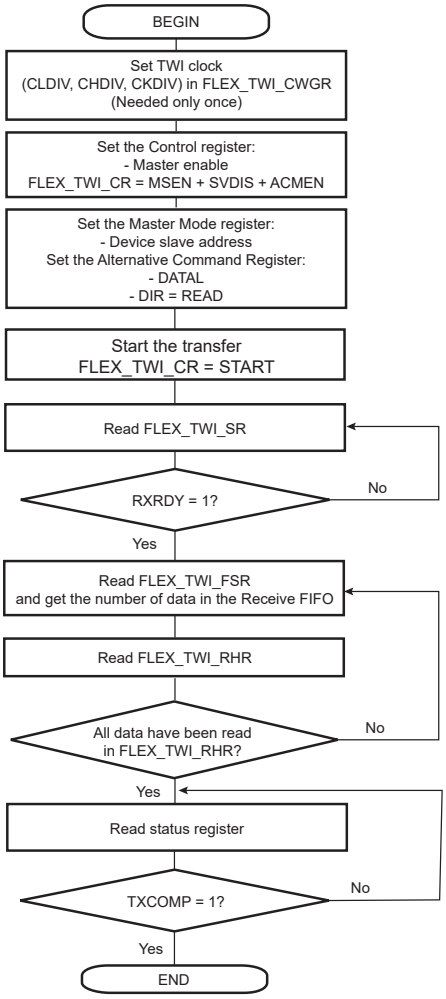

### 46.9.6.3 Receiving Data with FIFO Enabled

When the Receive FIFO is enabled, FLEX_TWI_RHR access reads the FIFO.

When data are present in the Receive FIFO (RXRDY flag set to ‘1’), the exact number of data can be checked with FLEX_TWI_FLR.RXFL. All the data can be read successively in FLEX_TWI_RHR without checking the FLEX_TWI_SR.RXRDY flag between each access.

See figures Receiving Data with FIFO Enabled in Master Mode and Sending/Receiving Data with FIFO Enabled in Slave Mode.

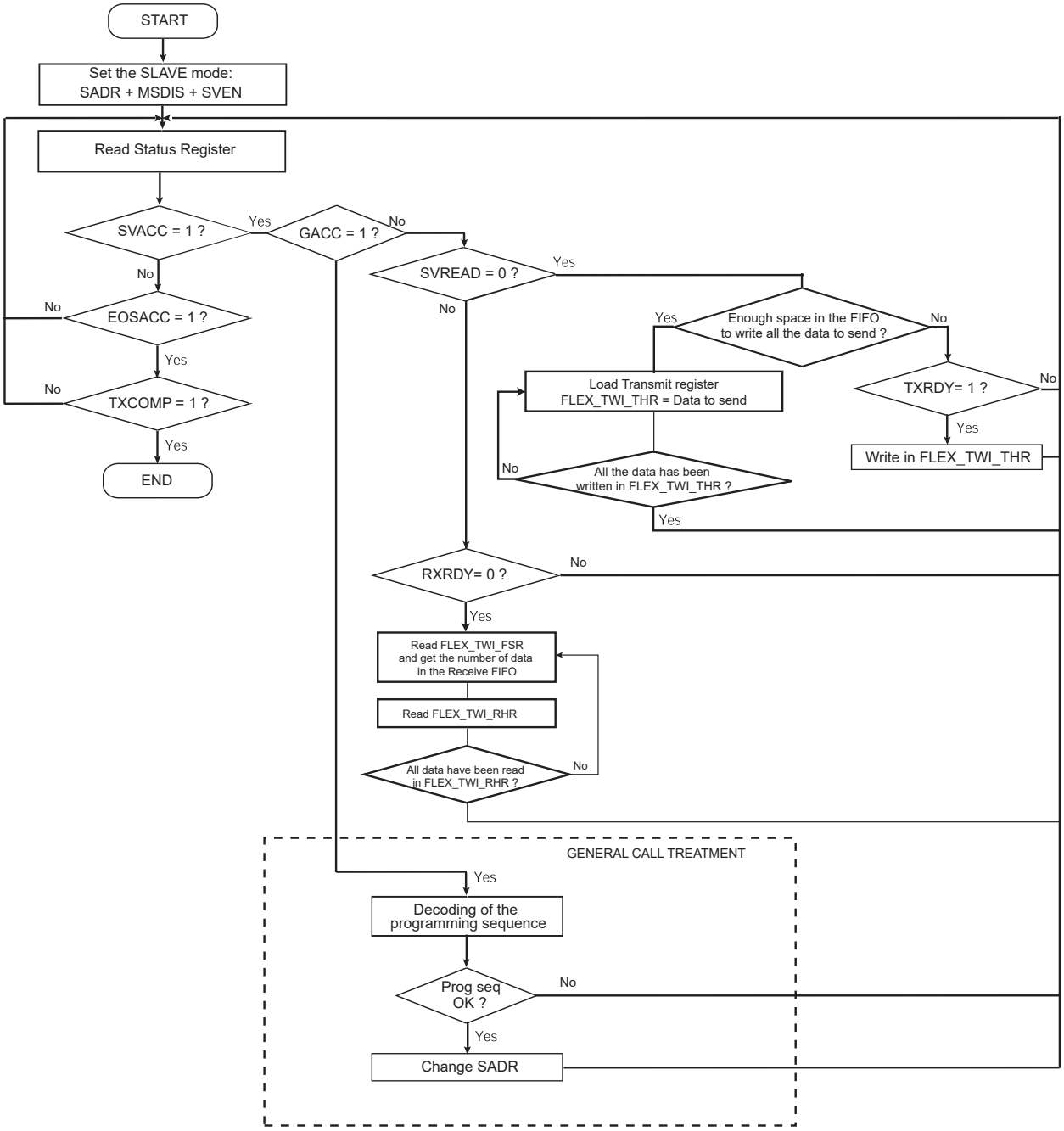
**Figure 46-129. Receiving Data with FIFO Enabled in Master Mode**




### 46.9.6.4 Sending/Receiving with FIFO Enabled in Slave Mode

See sections Sending Data with FIFO Enabled and Receiving Data with FIFO Enabled for details.

**Figure 46-130. Sending/Receiving Data with FIFO Enabled in Slave Mode**




### 46.9.6.5 Clearing/Flushing FIFOs

Each FIFO can be cleared/flushed using FLEX_TWI_CR.TXFCLR/RXFCLR.

### 46.9.6.6 TXRDY and RXRDY Behavior

FLEX_TWI_SR.TXRDY/RXRDY flags display a specific behavior when FIFOs are enabled.

TXRDY indicates if a data can be written in the Transmit FIFO. Thus the TXRDY flag is set as long as the Transmit FIFO can accept new data. See figure TXRDY Behavior when TXRDYM = 0 in Master mode.

RXRDY indicates if an unread data is present in the Receive FIFO. Thus the RXRDY flag is set as soon as one unread data is in the Receive FIFO. Refer to figure RXRDY Behavior when RXRDYM = 0 in Master and Slave modes.

TXRDY and RXRDY behavior can be modified using the TXRDYM and RXRDYM fields in the TWI FIFO Mode Register (FLEX_TWI_FMR) to reduce the number of accesses to FLEX_TWI_THR/RHR.

As an example, in Master mode, the Transmit FIFO can be loaded with multiple data in the same access by configuring TXRDYM>0.

See TWI FIFO Mode Register for the FIFO configuration.

**Figure 46-131. TXRDY Behavior when TXRDYM = 0 in Master Mode**

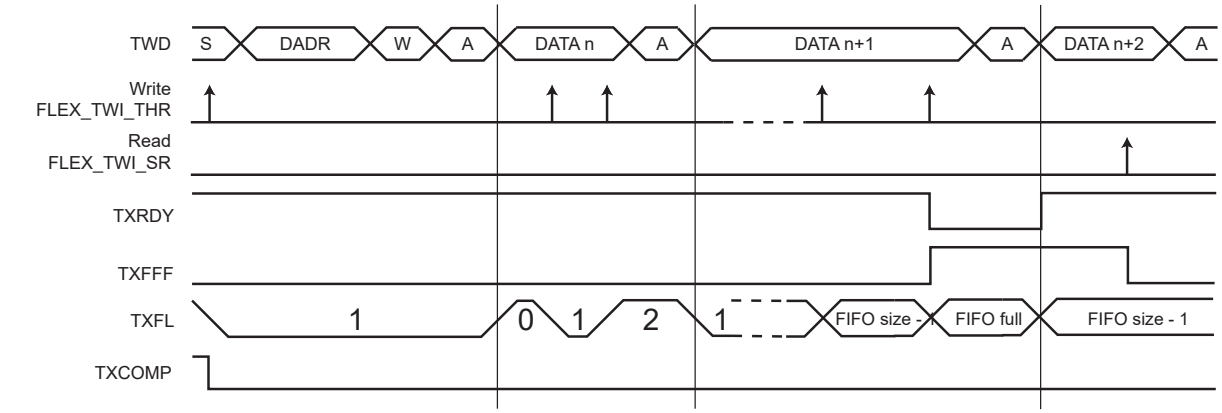



**Figure 46-132. RXRDY Behavior when RXRDYM = 0 in Master and Slave Modes**

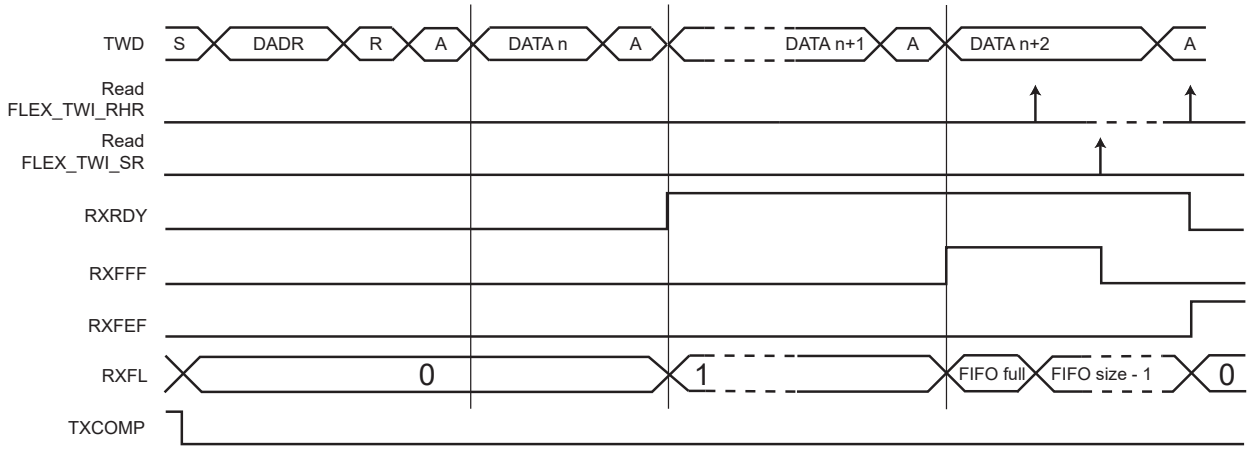
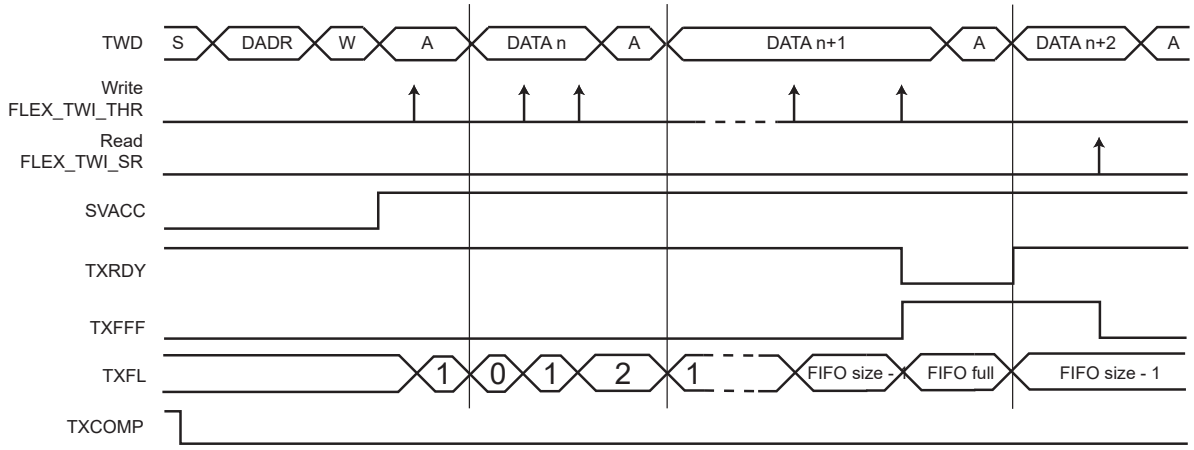


**Figure 46-133. TXRDY Behavior when TXRDYM = 0 in Slave Mode**




### 46.9.6.7 TWI Single Data Mode

In Single Data mode, only one data is written every time FLEX_TWI_THR is accessed, and only one data is read every time FLEX_TWI_RHR is accessed.

When FLEX_TWI_FMR.TXRDYM = 0, the Transmit FIFO operates in Single Data mode.

When FLEX_TWI_FMR.RXRDYM = 0, the Receive FIFO operates in Single Data mode.

See sections TWI Transmit Holding Register and TWI Receive Holding Register.

### 46.9.6.8 TWI Multiple Data Mode

Multiple Data mode minimizes the number of accesses by concatenating the data to send/read in one access.

When FLEX_TWI_FMR.TXRDYM > 0, the Transmit FIFO operates in Multiple Data mode.

When FLEX_TWI_FMR.RXRDYM > 0, the Receive FIFO operates in Multiple Data mode.

In Multiple Data mode, it is possible to write/read up to four data in one FLEX_TWI_THR/ FLEX_TWI_RHR register access.

The number of data to write/read is defined by the size of the register access. If the access is a byte-size register access, only one data is written/read. If the access is a halfword size register access, then up to two data are read and only one data is written. Lastly, if the access is a wordsize register access, then up to four data are read and up to two data are written.

Written/Read data are always right-aligned, as described in sections TWI Receive Holding Register (FIFO Enabled) and TWI Transmit Holding Register (FIFO Enabled).

As an example, if the Transmit FIFO is empty and there are six data to send, either of the following write accesses may be performed:

- six FLEX_TWI_THR-byte write accesses
- three FLEX_TWI_THR-halfword write accesses
- one FLEX_TWI_THR-word write access and one FLEX_TWI_THR halfword write access

With a Receive FIFO containing six data, any of the following read accesses may be performed:

- six FLEX_TWI_RHR-byte read accesses
- three FLEX_TWI_RHR-halfword read accesses
- one FLEX_TWI_RHR-word read access and one FLEX_TWI_RHR-halfword read access

#### 46.9.6.8.1 TXRDY and RXRDY Configuration

In Multiple Data mode, it is possible to write one or more data in the same FLEX_TWI_THR/FLEX_TWI_RHR access. The TXRDY flag indicates if one or more data can be written in the FIFO depending on the configuration of FLEX_TWI_FMR.TXRDYM/RXRDYM.

As an example, if two data are written each time in FLEX_TWI_THR, it is useful to configure the TXRDYM field to the value ‘1’ so that the TXRDY flag is at ‘1’ only when at least two data can be written in the Transmit FIFO.

In the same way, if four data are read each time in FLEX_TWI_RHR, it is useful to configure the RXRDYM field to the value ‘2’ so that the RXRDY flag is at ‘1’ only when at least four unread data are in the Receive FIFO.

#### 46.9.6.8.2 DMAC

When FIFOs operate in Multiple Data mode, the DMAC transfer type must be configured in byte, halfword or word depending on the FLEX_TWI_FMR.TXRDYM/RXRDYM settings.

### 46.9.6.9 Transmit FIFO Lock

If a frame is terminated early due to a not-acknowledge error (NACK flag), SMBus timeout error (TOUT flag) or master code acknowledge error (MACK flag), a lock is set on the Transmit FIFO preventing any new frame from being sent until it is cleared. This allows clearing the FIFO if needed, resetting DMAC channels, etc., without any risk.

FLEX_TWI_SR.LOCK is used to check the state of the Transmit FIFO lock.

The Transmit FIFO lock can be cleared by setting FLEX_TWI_CR.TXFLCLR to ‘1’.

### 46.9.6.10 FIFO Pointer Error

A FIFO overflow is reported in FLEX_TWI_FSR.

If the Transmit FIFO is full and a write access is performed on FLEX_TWI_THR, it generates a Transmit FIFO pointer error and sets FLEX_TWI_FSR.TXFPTEF.

In Multiple Data mode, if the number of data written in FLEX_TWI_THR (according to the register access size) is greater than the free space in the Transmit FIFO, a Transmit FIFO pointer error is generated and FLEX_TWI_FSR.TXFPTEF is set.

A FIFO underflow is reported in FLEX_TWI_FSR.

In Multiple Data mode, if the number of data read in FLEX_TWI_RHR (according to the register access size) is greater than the number of unread data in the Receive FIFO, a Receive FIFO pointer error is generated and FLEX_TWI_FSR.RXFPTEF is set.

No pointer error occurs if the FIFO state/level is checked before writing/reading in FLEX_TWI_THR/FLEX_TWI_RHR. The FIFO state/level can be checked either with TXRDY, RXRDY, TXFL or RXFL. When a pointer error occurs, other FIFO flags may not behave as expected; their states should be ignored.

If a Transmit or Receive pointer error occurs, a software reset must be performed using FLEX_TWI_CR.SWRST. Note that issuing a software while transmitting might leave a slave in an unknown state holding the TWD line. In such case, a Bus Clear Command will allow to make the slave release the TWD line (the first frame sent afterwards might not be received properly by the slave).

### 46.9.6.11 FIFO Thresholds

Each Transmit and Receive FIFO includes a threshold feature used to set a flag and an interrupt when a FIFO threshold is crossed. Thresholds are defined as a number of data in the FIFO, and the FIFO state (TXFL or RXFL) represents the number of data currently in the FIFO.

The Transmit FIFO threshold can be set using the field FLEX_TWI_FMR.TXFTHRES. Each time the Transmit FIFO level goes from ‘above threshold’ to ‘equal to or below threshold’, the flag FLEX_TWI_FESR.TXFTHF is set. The application is warned that the Transmit FIFO has reached the defined threshold and that it can be reloaded.

The Receive FIFO threshold can be set using the field FLEX_TWI_FMR.RXFTHRES. Each time the Receive FIFO level goes from ‘below threshold’ to ‘equal to or above threshold’, the flag FLEX_TWI_FESR.RXFTHF is set. The application is warned that the Receive FIFO has reached the defined threshold and that it can be read to prevent an underflow.

The TXFTHF and RXFTHF flags can be configured to generate an interrupt using FLEX_TWI_FIER and FLEX_TWI_FIDR.

#### 46.9.6.12 FIFO Flags

FIFOs come with a set of flags which can be configured to generate interrupts through FLEX_TWI_FIER and FLEX_TWI_FIDR.

FIFO flags state can be read in FLEX_TWI_FSR. They are cleared when FLEX_TWI_FSR is read.

### 46.9.7 TWI Comparison Function on Received Character

The comparison function differs if the asynchronous partial wakeup (Sleepwalking) is enabled or not.

If asynchronous partial wakeup is disabled (refer to the section “Power Management Controller” (PMC)), the TWI has the capability to extend the address matching on up to three slave addresses. The FLEX_TWI_SMR.SADR1EN/SADR2EN/SADR3EN bits enable address matching on additional addresses which can be configured through the FLEX_TWI_SWMR.SADR1/SADR2/SADR3 fields. The DATAMEN bit has no effect.

The SVACC bit is set when there is a comparison match with the received slave address.

### 46.9.8 TWI Register Write Protection

The FLEXCOM operating mode (FLEX_MR.OPMODE) must be set to FLEX_MR_OPMODE_TWI to enable access to the write protection registers.

To prevent any single software error from corrupting TWI behavior, certain registers in the address space can be write-protected by setting the WPEN (Write Protection Enable), WPITEN (Write Protection Interrupt Enable), and/or WPCREN (Write Protection Control Enable) bits in the TWI Write Protection Mode Register (FLEX_TWI_WPMR).

If a write access to a write-protected register is detected, the Write Protection Violation Status (WPVS) flag in the TWI Write Protection Status Register (FLEX_TWI_WPSR) is set and the Write Protection Violation Source (WPVSRC) field indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading FLEX_TWI_WPSR.

The following register(s) can be write-protected when WPEN is set:

- TWI Slave Mode Register
- TWI Clock Waveform Generator Register
- TWI SMBus Timing Register
- TWI SleepWalking Matching Register
- TWI FIFO Mode Register

The following register(s) can be write-protected when WPITEN is set:

- TWI Interrupt Enable Register

- TWI Interrupt Disable Register

The following register(s) can be write-protected when WPCREN is set:

- TWI Control Register

## 46.10 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | FLEX_MR | 7:0 | | | | | | | OPMODE[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | FLEX_RHR | 7:0 | RXDATA[7:0] | | | | | | | |
| | | 15:8 | RXDATA[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 ... 0x1F | Reserved | | | | | | | | | |
| 0x20 | FLEX_THR | 7:0 | TXDATA[7:0] | | | | | | | |
| | | 15:8 | TXDATA[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 ... 0x01FF | Reserved | | | | | | | | | |
| 0x0200 | FLEX_US_CR | 7:0 | TXDIS | TXEN | RXDIS | RXEN | RSTTX | RSTRX | | |
| | | 15:8 | RETTO | RSTNACK | RSTIT | SENDA | STTTO | STPBRK | STTBRK | RSTSTA |
| | | 23:16 | | | LINWKUP | LINABT | RTSDIS | RTSEN | | |
| | | 31:24 | FIFODIS | FIFOEN | | REQCLR | | TXFLCLR | RXFCLR | TXFCLR |
| 0x0200 | FLEX_US_CR (SPI_MODE) | 7:0 | TXDIS | TXEN | RXDIS | RXEN | RSTTX | RSTRX | | |
| | | 15:8 | | | | | | | | RSTSTA |
| | | 23:16 | | | | | RCS | FCS | | |
| | | 31:24 | | | | | | | | |
| 0x0204 | FLEX_US_MR | 7:0 | CHRL[1:0] | | USCLKS[1:0] | | USART_MODE[3:0] | | | |
| | | 15:8 | CHMODE[1:0] | | NBSTOP[1:0] | | PAR[2:0] | | | SYNC |
| | | 23:16 | INVDATA | VAR_SYNC | DSNACK | INACK | OVER | CLKO | MODE9 | MSBF |
| | | 31:24 | ONEBIT | MODSYNC | MAN | FILTER | | MAX_ITERATION[2:0] | | |
| 0x0204 | FLEX_US_MR (SPI_MODE) | 7:0 | CHRL[1:0] | | USCLKS[1:0] | | USART_MODE[3:0] | | | |
| | | 15:8 | CHMODE[1:0] | | | | | | | CPHA |
| | | 23:16 | | | | WRDBT | | | | CPOL |
| | | 31:24 | | | | | | | | |
| 0x0208 | FLEX_US_IER | 7:0 | PARE | FRAME | OVRE | | | RXBRK | TXRDY | RXRDY |
| | | 15:8 | | | NACK | | | ITER | TXEMPTY | TIMEOUT |
| | | 23:16 | | CMP | | | CTSIC | | | |
| | | 31:24 | | | | | | | | MANE |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0208 | FLEX_US_IER (SPI_MODE) | 7:0 | | | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | | | | | | UNRE | TXEMPTY | |
| | | 23:16 | | CMP | | | NSSE | | | |
| | | 31:24 | | | | | | | | |
| 0x0208 | FLEX_US_IER (LIN_MODE) | 7:0 | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | LINTC | LINID | LINBK | | | | TXEMPTY | TIMEOUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | LINHTE | LINSTE | LINSNRE | LINCE | LINIPE | LINISFE | LINBE | |
| 0x020C | FLEX_US_IDR | 7:0 | PARE | FRAME | OVRE | | | RXBRK | TXRDY | RXRDY |
| | | 15:8 | | | NACK | | | ITER | TXEMPTY | TIMEOUT |
| | | 23:16 | | CMP | | | CTSIC | | | |
| | | 31:24 | | | | | | | | MANE |
| 0x020C | FLEX_US_IDR (SPI_MODE) | 7:0 | | | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | | | | | | UNRE | TXEMPTY | |
| | | 23:16 | | CMP | | | NSSE | | | |
| | | 31:24 | | | | | | | | |
| 0x020C | FLEX_US_IDR (LIN_MODE) | 7:0 | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | LINTC | LINID | LINBK | | | | TXEMPTY | TIMEOUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | LINHTE | LINSTE | LINSNRE | LINCE | LINIPE | LINISFE | LINBE | |
| 0x0210 | FLEX_US_IMR | 7:0 | PARE | FRAME | OVRE | | | RXBRK | TXRDY | RXRDY |
| | | 15:8 | | | NACK | | | ITER | TXEMPTY | TIMEOUT |
| | | 23:16 | | CMP | | | CTSIC | | | |
| | | 31:24 | | | | | | | | MANE |
| 0x0210 | FLEX_US_IMR (SPI_MODE) | 7:0 | | | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | | | | | | UNRE | TXEMPTY | |
| | | 23:16 | | CMP | | | NSSE | | | |
| | | 31:24 | | | | | | | | |
| 0x0210 | FLEX_US_IMR (LIN_MODE) | 7:0 | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | LINTC | LINID | LINBK | | | | TXEMPTY | TIMEOUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | LINHTE | LINSTE | LINSNRE | LINCE | LINIPE | LINISFE | LINBE | |
| 0x0214 | FLEX_US_CSR | 7:0 | PARE | FRAME | OVRE | | | RXBRK | TXRDY | RXRDY |
| | | 15:8 | | | NACK | | | ITER | TXEMPTY | TIMEOUT |
| | | 23:16 | CTS | CMP | | | CTSIC | | | |
| | | 31:24 | | | | | | | | MANE |
| 0x0214 | FLEX_US_CSR (SPI_MODE) | 7:0 | | | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | | | | | | UNRE | TXEMPTY | |
| | | 23:16 | NSS | CMP | | | NSSE | | | |
| | | 31:24 | | | | | | | | |
| 0x0214 | FLEX_US_CSR (LIN_MODE) | 7:0 | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | LINTC | LINID | LINBK | | | | TXEMPTY | TIMEOUT |
| | | 23:16 | LINBLS | | | | | | | |
| | | 31:24 | LINHTE | LINSTE | LINSNRE | LINCE | LINIPE | LINISFE | LINBE | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0218 | FLEX_US_RHR | 7:0 | RXCHR[7:0] | | | | | | | |
| | | 15:8 | RXSYNH | | | | | | | RXCHR[8:8] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0218 | FLEX_US_RHR (FIFO_MULTI_DATA) | 7:0 | RXCHR0[7:0] | | | | | | | |
| | | 15:8 | RXCHR1[7:0] | | | | | | | |
| | | 23:16 | RXCHR2[7:0] | | | | | | | |
| | | 31:24 | RXCHR3[7:0] | | | | | | | |
| 0x021C | FLEX_US_THR | 7:0 | TXCHR[7:0] | | | | | | | |
| | | 15:8 | TXSYNH | | | | | | | TXCHR[8:8] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x021C | FLEX_US_THR (FIFO_MULTI_DATA) | 7:0 | TXCHR0[7:0] | | | | | | | |
| | | 15:8 | TXCHR1[7:0] | | | | | | | |
| | | 23:16 | TXCHR2[7:0] | | | | | | | |
| | | 31:24 | TXCHR3[7:0] | | | | | | | |
| 0x0220 | FLEX_US_BRGR | 7:0 | CD[7:0] | | | | | | | |
| | | 15:8 | CD[15:8] | | | | | | | |
| | | 23:16 | | | | | | FP[2:0] | | |
| | | 31:24 | | | | | | | | |
| 0x0224 | FLEX_US_RTOR | 7:0 | TO[7:0] | | | | | | | |
| | | 15:8 | TO[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | TO[16:16] |
| | | 31:24 | | | | | | | | |
| 0x0228 | FLEX_US_TTGR | 7:0 | TG[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x022C ... 0x023F | Reserved | | | | | | | | | |
| 0x0240 | FLEX_US_FIDI | 7:0 | FI_DI_RATIO[7:0] | | | | | | | |
| | | 15:8 | FI_DI_RATIO[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0244 | FLEX_US_NER | 7:0 | NB_ERRORS[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0248 ... 0x024B | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x024C | FLEX_US_IF | 7:0 | IRDA_FILTER[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0250 | FLEX_US_MAN | 7:0 | | | | | TX_PL[3:0] | | | |
| | | 15:8 | | | | TX_MPOL | | | TX_PP[1:0] | |
| | | 23:16 | | | | | RX_PL[3:0] | | | |
| | | 31:24 | RXIDLEV | DRIFT | ONE | RX_MPOL | | | RX_PP[1:0] | |
| 0x0254 | FLEX_US_LINMR | 7:0 | WKUPTYP | FSDIS | DLM | CHKTYP | CHKDIS | PARDIS | NACT[1:0] | |
| | | 15:8 | DLC[7:0] | | | | | | | |
| | | 23:16 | | | | | | | SYNCDIS | PDCM |
| | | 31:24 | | | | | | | | |
| 0x0258 | FLEX_US_LINIR | 7:0 | IDCHR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x025C | FLEX_US_LINBRR | 7:0 | LINCD[7:0] | | | | | | | |
| | | 15:8 | LINCD[15:8] | | | | | | | |
| | | 23:16 | | | | | | LINFP[2:0] | | |
| | | 31:24 | | | | | | | | |
| 0x0260 ... 0x028F | Reserved | | | | | | | | | |
| 0x0290 | FLEX_US_CMPR | 7:0 | VAL1[7:0] | | | | | | | |
| | | 15:8 | | CMPPAR | | CMPMODE | | | | VAL1[8:8] |
| | | 23:16 | VAL2[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | VAL2[8:8] |
| 0x0294 ... 0x029F | Reserved | | | | | | | | | |
| 0x02A0 | FLEX_US_FMR | 7:0 | FRTSC | | RXRDYM[1:0] | | | | TXRDYM[1:0] | |
| | | 15:8 | | | TXFTHRES[5:0] | | | | | |
| | | 23:16 | | | RXFTHRES[5:0] | | | | | |
| | | 31:24 | | | RXFTHRES2[5:0] | | | | | |
| 0x02A4 | FLEX_US_FLR | 7:0 | | | TXFL[5:0] | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | RXFL[5:0] | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02A8 | FLEX_US_FIER | 7:0 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| | | 15:8 | | | | | | | RXFTHF2 | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x02AC | FLEX_US_FIDR | 7:0 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| | | 15:8 | | | | | | | RXFTHF2 | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02B0 | FLEX_US_FIMR | 7:0 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| | | 15:8 | | | | | | | RXFTHF2 | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02B4 | FLEX_US_FESR | 7:0 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| | | 15:8 | | | | | | | RXFTHF2 | TXFLOCK |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02B8 ... 0x02E3 | Reserved | | | | | | | | | |
| 0x02E4 | FLEX_US_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0x02E8 | FLEX_US_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02EC ... 0x03FF | Reserved | | | | | | | | | |
| 0x0400 | FLEX_SPI_CR | 7:0 | SWRST | | | | | | SPIDIS | SPIEN |
| | | 15:8 | | | | REQCLR | | | | |
| | | 23:16 | | | | | | | RXFCLR | TXFCLR |
| | | 31:24 | FIFODIS | FIFOEN | | | | | | LASTXFER |
| 0x0404 | FLEX_SPI_MR | 7:0 | LLB | | WDRBT | MODFDIS | BRSRCCLK | PCSDEC | PS | MSTR |
| | | 15:8 | | | | CMPMODE | | | | |
| | | 23:16 | | | | | | | PCS[1:0] | |
| | | 31:24 | DLYBCS[7:0] | | | | | | | |
| 0x0408 | FLEX_SPI_RDR | 7:0 | RD[7:0] | | | | | | | |
| | | 15:8 | RD[15:8] | | | | | | | |
| | | 23:16 | | | | | PCS[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x0408 | FLEX_SPI_RDR (FIFO_MULTI_DATA_8) | 7:0 | RD0[7:0] | | | | | | | |
| | | 15:8 | RD1[7:0] | | | | | | | |
| | | 23:16 | RD2[7:0] | | | | | | | |
| | | 31:24 | RD3[7:0] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0408 | FLEX_SPI_RDR (FIFO_MULTI_DATA_16) | 7:0 | RD0[7:0] | | | | | | | |
| | | 15:8 | RD0[15:8] | | | | | | | |
| | | 23:16 | RD1[7:0] | | | | | | | |
| | | 31:24 | RD1[15:8] | | | | | | | |
| 0x040C | FLEX_SPI_TDR | 7:0 | TD[7:0] | | | | | | | |
| | | 15:8 | TD[15:8] | | | | | | | |
| | | 23:16 | | | | | PCS[3:0] | | | |
| | | 31:24 | | | | | | | | LASTXFER |
| 0x040C | FLEX_SPI_TDR (FIFO_MULTI_DATA) | 7:0 | TD0[7:0] | | | | | | | |
| | | 15:8 | TD0[15:8] | | | | | | | |
| | | 23:16 | TD1[7:0] | | | | | | | |
| | | 31:24 | TD1[15:8] | | | | | | | |
| 0x0410 | FLEX_SPI_SR | 7:0 | | | | | OVRES | MODF | TDRE | RDRF |
| | | 15:8 | | | | | CMP | UNDES | TXEMPTY | NSSR |
| | | 23:16 | | | | | | | | SPIENS |
| | | 31:24 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| 0x0414 | FLEX_SPI_IER | 7:0 | | | | | OVRES | MODF | TDRE | RDRF |
| | | 15:8 | | | | | CMP | UNDES | TXEMPTY | NSSR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| 0x0418 | FLEX_SPI_IDR | 7:0 | | | | | OVRES | MODF | TDRE | RDRF |
| | | 15:8 | | | | | CMP | UNDES | TXEMPTY | NSSR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| 0x041C | FLEX_SPI_IMR | 7:0 | | | | | OVRES | MODF | TDRE | RDRF |
| | | 15:8 | | | | | CMP | UNDES | TXEMPTY | NSSR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| 0x0420 ... 0x042F | Reserved | | | | | | | | | |
| 0x0430 | FLEX_SPI_CSR0 | 7:0 | BITS[3:0] | | | | CSAAT | CSNAAT | NCPHA | CPOL |
| | | 15:8 | SCBR[7:0] | | | | | | | |
| | | 23:16 | DLYBS[7:0] | | | | | | | |
| | | 31:24 | DLYBCT[7:0] | | | | | | | |
| 0x0434 | FLEX_SPI_CSR1 | 7:0 | BITS[3:0] | | | | CSAAT | CSNAAT | NCPHA | CPOL |
| | | 15:8 | SCBR[7:0] | | | | | | | |
| | | 23:16 | DLYBS[7:0] | | | | | | | |
| | | 31:24 | DLYBCT[7:0] | | | | | | | |
| 0x0438 ... 0x043F | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0440 | FLEX_SPI_FMR | 7:0 | | | RXRDYM[1:0] | | | | TXRDYM[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | TXFTHRES[5:0] | | | | | |
| | | 31:24 | | | RXFTHRES[5:0] | | | | | |
| 0x0444 | FLEX_SPI_FLR | 7:0 | | | TXFL[5:0] | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | RXFL[5:0] | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0448 | FLEX_SPI_CMPR | 7:0 | VAL1[7:0] | | | | | | | |
| | | 15:8 | VAL1[15:8] | | | | | | | |
| | | 23:16 | VAL2[7:0] | | | | | | | |
| | | 31:24 | VAL2[15:8] | | | | | | | |
| 0x044C ... 0x04E3 | Reserved | | | | | | | | | |
| 0x04E4 | FLEX_SPI_WPMR | 7:0 | | | | | | WPCREN | WPITEN | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0x04E8 | FLEX_SPI_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04EC ... 0x05FF | Reserved | | | | | | | | | |
| 0x0600 | FLEX_TWI_CR | 7:0 | SWRST | QUICK | SVDIS | SVEN | MSDIS | MSEN | STOP | START |
| | | 15:8 | CLEAR | PECRQ | PECDIS | PECEN | SMBDIS | SMBEN | HSDIS | HSEN |
| | | 23:16 | | | | | | | ACMDIS | ACMEN |
| | | 31:24 | | | FIFODIS | FIFOEN | | LOCKCLR | | THRCLR |
| 0x0600 | FLEX_TWI_CR (FIFO_ENABLED) | 7:0 | SWRST | QUICK | SVDIS | SVEN | MSDIS | MSEN | STOP | START |
| | | 15:8 | CLEAR | PECRQ | PECDIS | PECEN | SMBDIS | SMBEN | HSDIS | HSEN |
| | | 23:16 | | | | | | | ACMDIS | ACMEN |
| | | 31:24 | | | FIFODIS | FIFOEN | | TXFLCLR | RXFCLR | TXFCLR |
| 0x0604 | FLEX_TWI_MMR | 7:0 | | | | | | | | |
| | | 15:8 | | | | MREAD | | | IADRSZ[1:0] | |
| | | 23:16 | | DADR[6:0] | | | | | | |
| | | 31:24 | | | | | | | | NOAP |
| 0x0608 | FLEX_TWI_SMR | 7:0 | | SCLWSDIS | | SADAT | SMHH | SMDA | | NACKEN |
| | | 15:8 | | MASK[6:0] | | | | | | |
| | | 23:16 | | SADR[6:0] | | | | | | |
| | | 31:24 | DATAMEN | SADR3EN | SADR2EN | SADR1EN | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x060C | FLEX_TWI_IADR | 7:0 | IADR[7:0] | | | | | | | |
| | | 15:8 | IADR[15:8] | | | | | | | |
| | | 23:16 | IADR[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0610 | FLEX_TWI_CWGR | 7:0 | CLDIV[7:0] | | | | | | | |
| | | 15:8 | CHDIV[7:0] | | | | | | | |
| | | 23:16 | | | | BRSRCCLK | | CKDIV[2:0] | | |
| | | 31:24 | | | HOLD[5:0] | | | | | |
| 0x0614 ... 0x061F | Reserved | | | | | | | | | |
| 0x0620 | FLEX_TWI_SR | 7:0 | UNRE | OVRE | GACC | SVACC | SVREAD | TXRDY | RXRDY | TXCOMP |
| | | 15:8 | | | | | EOSACC | SCLWS | ARBLST | NACK |
| | | 23:16 | LOCK | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| | | 31:24 | | | | | | SR | SDA | SCL |
| 0x0620 | FLEX_TWI_SR (FIFO_ENABLED) | 7:0 | UNRE | OVRE | GACC | SVACC | SVREAD | TXRDY | RXRDY | TXCOMP |
| | | 15:8 | | | | | EOSACC | SCLWS | ARBLST | NACK |
| | | 23:16 | TXFLOCK | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| | | 31:24 | | | | | | SR | SDA | SCL |
| 0x0624 | FLEX_TWI_IER | 7:0 | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| | | 15:8 | TXBUFE | RXBUFF | ENDTX | ENDRX | EOSACC | SCL_WS | ARBLST | NACK |
| | | 23:16 | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| | | 31:24 | | | | | | | | |
| 0x0628 | FLEX_TWI_IDR | 7:0 | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| | | 15:8 | TXBUFE | RXBUFF | ENDTX | ENDRX | EOSACC | SCL_WS | ARBLST | NACK |
| | | 23:16 | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| | | 31:24 | | | | | | | | |
| 0x062C | FLEX_TWI_IMR | 7:0 | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| | | 15:8 | TXBUFE | RXBUFF | ENDTX | ENDRX | EOSACC | SCL_WS | ARBLST | NACK |
| | | 23:16 | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| | | 31:24 | | | | | | | | |
| 0x0630 | FLEX_TWI_RHR | 7:0 | RXDATA[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0630 | FLEX_TWI_RHR (FIFO_ENABLED) | 7:0 | RXDATA0[7:0] | | | | | | | |
| | | 15:8 | RXDATA1[7:0] | | | | | | | |
| | | 23:16 | RXDATA2[7:0] | | | | | | | |
| | | 31:24 | RXDATA3[7:0] | | | | | | | |
| 0x0634 | FLEX_TWI_THR | 7:0 | TXDATA[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0634 | FLEX_TWI_THR (FIFO_ENABLED) | 7:0 | TXDATA0[7:0] | | | | | | | |
| | | 15:8 | TXDATA1[7:0] | | | | | | | |
| | | 23:16 | TXDATA2[7:0] | | | | | | | |
| | | 31:24 | TXDATA3[7:0] | | | | | | | |
| 0x0638 | FLEX_TWI_SMBTR | 7:0 | | | | | PRESC[3:0] | | | |
| | | 15:8 | TLOWS[7:0] | | | | | | | |
| | | 23:16 | TLOWM[7:0] | | | | | | | |
| | | 31:24 | THMAX[7:0] | | | | | | | |
| 0x063C ... 0x063F | Reserved | | | | | | | | | |
| 0x0640 | FLEX_TWI_ACR | 7:0 | DATAL[7:0] | | | | | | | |
| | | 15:8 | | | | | | | PEC | DIR |
| | | 23:16 | NDATAL[7:0] | | | | | | | |
| | | 31:24 | | | | | | | NPEC | NDIR |
| 0x0644 | FLEX_TWI_FILTR | 7:0 | | | | | | PADFCFG | PADFEN | FILT |
| | | 15:8 | | | | | | THRES[2:0] | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0648 ... 0x064B | Reserved | | | | | | | | | |
| 0x064C | FLEX_TWI_SWMR | 7:0 | | SADR1[6:0] | | | | | | |
| | | 15:8 | | SADR2[6:0] | | | | | | |
| | | 23:16 | | SADR3[6:0] | | | | | | |
| | | 31:24 | DATAM[7:0] | | | | | | | |
| 0x0650 | FLEX_TWI_FMR | 7:0 | | | RXRDYM[1:0] | | | | TXRDYM[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | TXFTHRES[5:0] | | | | | |
| | | 31:24 | | | RXFTHRES[5:0] | | | | | |
| 0x0654 | FLEX_TWI_FLR | 7:0 | | | TXFL[5:0] | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | RXFL[5:0] | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0658 ... 0x065F | Reserved | | | | | | | | | |
| 0x0660 | FLEX_TWI_FSR | 7:0 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0664 | FLEX_TWI_FIER | 7:0 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0668 | FLEX_TWI_FIDR | 7:0 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x066C | FLEX_TWI_FIMR | 7:0 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0670 ... 0x06E3 | Reserved | | | | | | | | | |
| 0x06E4 | FLEX_TWI_WPMR | 7:0 | | | | | | WPCREN | WPITEN | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0x06E8 | FLEX_TWI_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | WPVSRC[23:16] | | | | | | | |

### 46.10.1 FLEXCOM Mode Register

**Name:** FLEX_MR
**Offset:** 0x000
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | OPMODE[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bits 1:0 – OPMODE[1:0]** FLEXCOM Operating Mode

| Value | Name | Description |
|---|---|---|
| 0 | NO_COM | No communication |
| 1 | USART | All UART-related protocols are selected (RS232, RS485, IrDA, ISO7816, LIN,)<br>SPI/TWI-related registers are not accessible and have no impact on IOs. |
| 2 | SPI | SPI operating mode is selected.<br>USART/TWI related registers are not accessible and have no impact on IOs. |
| 3 | TWI | All TWI-related protocols are selected (TWI, SMBus).<br>USART/SPI-related registers are not accessible and have no impact on IOs. |

### 46.10.2 FLEXCOM Receive Holding Register

**Name:** FLEX_RHR
**Offset:** 0x010
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RXDATA[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXDATA[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RXDATA[15:0]** Receive Data
This register is a mirror of:

- USART Receive Holding Register (FLEX_US_RHR) if FLEX_MR.OPMODE field equals 1
- SPI Receive Data Register (FLEX_SPI_RDR) if FLEX_MR.OPMODE field equals 2
- TWI Transmit Holding Register (FLEX_TWI_RHR) if FLEX_MR.OPMODE field equals 3

### 46.10.3 FLEXCOM Transmit Holding Register

**Name:** FLEX_THR
**Offset:** 0x020
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TXDATA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXDATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – TXDATA[15:0]** Transmit Data
This register is a mirror of:

- USART Transmit Holding Register (FLEX_US_THR) if FLEX_MR.OPMODE field equals 1
- SPI Transmit Data Register (FLEX_SPI_TDR) if FLEX_MR.OPMODE field equals 2
- TWI Transmit Holding Register (FLEX_TWI_THR) if FLEX_MR.OPMODE field equals 3

### 46.10.4 USART Control Register

**Name:** FLEX_US_CR
**Offset:** 0x200
**Reset:** –
**Property:** Write-only

For SPI control, see USART Control Register (SPI_MODE).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FIFODIS | FIFOEN | | REQCLR | | TXFLCLR | RXFCLR | TXFCLR |
| Access | W | W | | W | | W | W | W |
| Reset | – | – | | – | | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | LINWKUP | LINABT | RTSDIS | RTSEN | | |
| Access | | | W | W | W | W | | |
| Reset | | | – | – | – | – | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RETTO | RSTNACK | RSTIT | SENDA | STTTO | STPBRK | STTBRK | RSTSTA |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXDIS | TXEN | RXDIS | RXEN | RSTTX | RSTRX | | |
| Access | W | W | W | W | W | W | | |
| Reset | – | – | – | – | – | – | | |

**Bit 31 – FIFODIS** FIFO Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Transmit and Receive FIFOs. |

**Bit 30 – FIFOEN** FIFO Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Transmit and Receive FIFOs. |

**Bit 28 – REQCLR** Request to Clear the Comparison Trigger

**Bit 26 – TXFLCLR** Transmit FIFO Lock CLEAR

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears the Transmit FIFO Lock. |

**Bit 25 – RXFCLR** Receive FIFO Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Empties the Receive FIFO. |

**Bit 24 – TXFCLR** Transmit FIFO Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Empties the Transmit FIFO. |

**Bit 21 – LINWKUP** Send LIN Wakeup Signal

| Value | Description |
|---|---|
| 0 | No effect: |
| 1 | Sends a wakeup signal on the LIN bus. |

**Bit 20 – LINABT** Abort LIN Transmission

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Aborts the current LIN transmission. |

**Bit 19 – RTSDIS** Request to Send Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Drives the RTS pin to 0 if FLEX_US_MR.USART_MODE field = 2, else drives the RTS pin to 1 if FLEX_US_MR.USART_MODE field = 0. |

**Bit 18 – RTSEN** Request to Send Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Drives the RTS pin to 1 if FLEX_US_MR.USART_MODE field = 2, else drives the RTS pin to 0 if FLEX_US_MR.USART_MODE field = 0. |

**Bit 15 – RETTO** Start Timeout Immediately

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Immediately restarts timeout period. |

**Bit 14 – RSTNACK** Reset Non Acknowledge

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Resets FLEX_US_CSR.NACK. |

**Bit 13 – RSTIT** Reset Iterations

| Value | Description |
|---|---|
| 0 | No effect. |

| Value | Description |
|---|---|
| 1 | Resets FLEX_US_CSR.ITER. No effect if the ISO7816 is not enabled. |

**Bit 12 – SENDA** Send Address

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | In Multidrop mode only, the next character written to FLEX_US_THR is sent with the address bit set. |

**Bit 11 – STTTO** Clear TIMEOUT Flag and Start Timeout After Next Character Received

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Starts waiting for a character before clocking the timeout counter. Immediately disables a timeout period in progress. Resets the FLEX_US_CSR.TIMEOUT status bit. |

**Bit 10 – STPBRK** Stop Break

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Stops transmission of the break after a minimum of one character length and transmits a high level during 12-bit periods. No effect if no break is being transmitted. |

**Bit 9 – STTBRK** Start Break

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Starts transmission of a break after the characters present in FLEX_US_THR and the Transmit Shift Register have been transmitted. No effect if a break is already being transmitted. |

**Bit 8 – RSTSTA** Reset Status Bits

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the PARE, FRAME, OVRE, MANE, LINBE, LINISFE, LINIPE, LINCE, LINSNRE, LINSTE, LINHTE, LINID, LINTC, LINBK, CMP and RXBRK in FLEX_US_CSR status bits, as well as the TXFEF, TXFFF, TXFTHF, RXFEF, RXFFF, RXFTHF, TXFPTEF, RXFPTEF in FLEX_US_FESR status bits. |

**Bit 7 – TXDIS** Transmitter Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the transmitter. |

**Bit 6 – TXEN** Transmitter Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the transmitter if TXDIS is 0. |

**Bit 5 – RXDIS** Receiver Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the receiver. |

**Bit 4 – RXEN** Receiver Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the receiver, if RXDIS is 0. |

**Bit 3 – RSTTX** Reset Transmitter

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the transmitter. |

**Bit 2 – RSTRX** Reset Receiver

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the receiver. |

### 46.10.5 USART Control Register (SPI_MODE)

**Name:** FLEX_US_CR (SPI_MODE)
**Offset:** 0x200
**Reset:** –
**Property:** Write-only

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RCS | FCS | | |
| Access | | | | | W | W | | |
| Reset | | | | | – | – | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | RSTSTA |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXDIS | TXEN | RXDIS | RXEN | RSTTX | RSTRX | | |
| Access | W | W | W | W | W | W | | |
| Reset | – | – | – | – | – | – | | |

**Bit 19 – RCS** Release SPI Chip Select
Applicable if USART operates in SPI Master mode (USART_MODE = 0xE):

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Releases the Slave Select Line NSS (RTS pin). |

**Bit 18 – FCS** Force SPI Chip Select
Applicable if USART operates in SPI Master mode (USART_MODE = 0xE):

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Forces the Slave Select Line NSS (RTS pin) to 0, even if USART is not transmitting, in order to address SPI slave devices supporting the CSAAT mode (Chip Select Active After Transfer). |

**Bit 8 – RSTSTA** Reset Status Bits

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the FLEX_US_CSR.OVRE/UNRE status bits. |

**Bit 7 – TXDIS** Transmitter Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the transmitter. |

**Bit 6 – TXEN** Transmitter Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the transmitter if TXDIS is 0. |

**Bit 5 – RXDIS** Receiver Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the receiver. |

**Bit 4 – RXEN** Receiver Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the receiver, if RXDIS is 0. |

**Bit 3 – RSTTX** Reset Transmitter

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the transmitter. |

**Bit 2 – RSTRX** Reset Receiver

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the receiver. |

### 46.10.6 USART Mode Register

**Name:** FLEX_US_MR
**Offset:** 0x204
**Reset:** –
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

For SPI configuration, see section USART Mode Register (SPI_MODE).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ONEBIT | MODSYNC | MAN | FILTER | | MAX_ITERATION[2:0] | | |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | – | – | – | – | | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | INVDATA | VAR_SYNC | DSNACK | INACK | OVER | CLKO | MODE9 | MSBF |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CHMODE[1:0] | | NBSTOP[1:0] | | PAR[2:0] | | | SYNC |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CHRL[1:0] | | USCLKS[1:0] | | USART_MODE[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 31 – ONEBIT** Start Frame Delimiter Selector

| Value | Description |
|---|---|
| 0 | Start frame delimiter is COMMAND or DATA SYNC. |
| 1 | Start frame delimiter is one bit. |

**Bit 30 – MODSYNC** Manchester Synchronization Mode

| Value | Description |
|---|---|
| 0 | The Manchester start bit is a 0 to 1 transition |
| 1 | The Manchester start bit is a 1 to 0 transition. |

**Bit 29 – MAN** Manchester Encoder/Decoder Enable

| Value | Description |
|---|---|
| 0 | Manchester encoder/decoder are disabled. |
| 1 | Manchester encoder/decoder are enabled. |

**Bit 28 – FILTER** Receive Line Filter

| Value | Description |
|---|---|
| 0 | The USART does not filter the receive line. |
| 1 | The USART filters the receive line using a three-sample filter (1/16-bit clock) (2 over 3 majority). |

**Bits 26:24 – MAX_ITERATION[2:0]** Maximum Number of Automatic Iterations

| Value | Description |
|---|---|
| 0-7 | Defines the maximum number of iterations in mode ISO7816, protocol T = 0. |

**Bit 23 – INVDATA** Inverted Data

| Value | Description |
|---|---|
| 0 | The data field transmitted on TXD line is the same as the one written in FLEX_US_THR or the content read in FLEX_US_RHR is the same as RXD line. Normal mode of operation. |
| 1 | The data field transmitted on TXD line is inverted (voltage polarity only) compared to the value written in FLEX_US_THR or the content read in FLEX_US_RHR is inverted compared to what is received on RXD line (or ISO7816 IO line). Inverted mode of operation, useful for contactless card application. To be used with configuration bit MSBF. |

**Bit 22 – VAR_SYNC** Variable Synchronization of Command/Data Sync Start Frame Delimiter

| Value | Description |
|---|---|
| 0 | User defined configuration of command or data sync field depending on MODSYNC value. |
| 1 | The sync field is updated when a character is written into FLEX_US_THR. |

**Bit 21 – DSNACK** Disable Successive NACK
The MAX_ITERATION field must be cleared if DSNACK is cleared.

| Value | Description |
|---|---|
| 0 | NACK is sent on the ISO line as soon as a parity error occurs in the received character (unless INACK is set). |
| 1 | Successive parity errors are counted up to the value specified in the MAX_ITERATION field. These parity errors generate a NACK on the ISO line. As soon as this value is reached, no additional NACK is sent on the ISO line. The flag ITER is asserted. |

**Bit 20 – INACK** Inhibit Non Acknowledge

| Value | Description |
|---|---|
| 0 | The NACK is generated. |
| 1 | The NACK is not generated. |

**Bit 19 – OVER** Oversampling Mode

| Value | Description |
|---|---|
| 0 | 16x Oversampling. |
| 1 | 8x Oversampling. |

**Bit 18 – CLKO** Clock Output Select

| Value | Description |
|---|---|
| 0 | The USART does not drive the SCK pin (Synchronous Slave mode or Asynchronous mode with external baud rate clock source). |
| 1 | The USART drives the SCK pin if USCLKS does not select the external clock SCK (USART Synchronous Master mode). |

**Bit 17 – MODE9** 9-bit Character Length

| Value | Description |
|---|---|
| 0 | CHRL defines character length. |
| 1 | 9-bit character length. |

**Bit 16 – MSBF** Bit Order

| Value | Description |
|---|---|
| 0 | Least significant bit is sent/received first. |
| 1 | Most significant bit is sent/received first. |

**Bits 15:14 – CHMODE[1:0]** Channel Mode

| Value | Name | Description |
|---|---|---|
| 0 | NORMAL | Normal mode |
| 1 | AUTOMATIC | Automatic Echo. Receiver input is connected to the TXD pin. |
| 2 | LOCAL_LOOPBACK | Local Loopback. Transmitter output is connected to the Receiver Input. |
| 3 | REMOTE_LOOPBACK | Remote Loopback. RXD pin is internally connected to the TXD pin. |

**Bits 13:12 – NBSTOP[1:0]** Number of Stop Bits

| Value | Name | Description |
|---|---|---|
| 0 | 1_BIT | 1 stop bit |
| 1 | 1_5_BIT | 1.5 stop bit (SYNC = 0) or reserved (SYNC = 1) |
| 2 | 2_BIT | 2 stop bits |

**Bits 11:9 – PAR[2:0]** Parity Type

| Value | Name | Description |
|---|---|---|
| 0 | EVEN | Even parity |
| 1 | ODD | Odd parity |
| 2 | SPACE | Parity forced to 0 (Space) |
| 3 | MARK | Parity forced to 1 (Mark) |
| 4 | NO | No parity |
| 6 | MULTIDROP | Multidrop mode |

**Bit 8 – SYNC** Synchronous Mode Select

| Value | Description |
|---|---|
| 0 | USART operates in Asynchronous mode (UART). |
| 1 | USART operates in Synchronous mode. |

**Bits 7:6 – CHRL[1:0]** Character Length

| Value | Name | Description |
|---|---|---|
| 0 | 5_BIT | Character length is 5 bits |
| 1 | 6_BIT | Character length is 6 bits |
| 2 | 7_BIT | Character length is 7 bits |
| 3 | 8_BIT | Character length is 8 bits |

**Bits 5:4 – USCLKS[1:0]** Clock Selection

| Value | Name | Description |
|---|---|---|
| 0 | MCK | Peripheral clock is selected |
| 1 | DIV | Peripheral clock divided (DIV = 8) is selected |
| 2 | GCLK | PMC generic clock is selected. If the SCK pin is driven (CLKO = 1), the CD field must be greater than 1. |
| 3 | SCK | External pin SCK is selected |

**Bits 3:0 – USART_MODE[3:0]** USART Mode of Operation

| Value | Name | Description |
|---|---|---|
| 0x0 | NORMAL | Normal mode |
| 0x1 | RS485 | RS485 |
| 0x2 | HW_HANDSHAKING | Hardware handshaking |
| 0x3 | MODEM | Modem |
| 0x4 | IS07816_T_0 | IS07816 Protocol: T = 0 |
| 0x6 | IS07816_T_1 | IS07816 Protocol: T = 1 |
| 0x8 | IRDA | IrDA |
| 0x9 | LON | LON |
| 0xA | LIN_MASTER | LIN Master mode |
| 0xB | LIN_SLAVE | LIN Slave mode |
| 0xC | DATA16BIT_MASTER | 16-bit data master |
| 0xD | DATA16BIT_SLAVE | 16-bit data slave |
| 0xE | SPI_MASTER | SPI Master mode (CLKO must be written to 1 and USCLKS = 0, 1 or 2) |
| 0xF | SPI_SLAVE | SPI Slave mode |

### 46.10.7 USART Mode Register (SPI_MODE)

**Name:** FLEX_US_MR (SPI_MODE)
**Offset:** 0x204
**Reset:** –
**Property:** Read/Write

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | WRDBT | | | | CPOL |
| Access | | | | R/W | | | | R/W |
| Reset | | | | – | | | | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CHMODE[1:0] | | | | | | | CPHA |
| Access | R/W | R/W | | | | | | R/W |
| Reset | 0 | – | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CHRL[1:0] | | USCLKS[1:0] | | USART_MODE[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | – | 0 | – | 0 | 0 | 0 | – |

**Bit 20 – WRDBT** Wait Read Data Before Transfer

| Value | Description |
|---|---|
| 0 | The character transmission starts as soon as a character is written into FLEX_US_THR (assuming TXRDY was set). |
| 1 | The character transmission starts when a character is written and only if RXRDY flag is cleared (Receive Holding Register has been read). |

**Bit 16 – CPOL** SPI Clock Polarity
CPOL is used to determine the inactive state value of the serial clock (SPCK). It is used with CPHA to produce the required clock/data relationship between master and slave devices.

Applicable if USART operates in SPI mode (slave or master, USART_MODE = 0xE or 0xF):

| Value | Description |
|---|---|
| 0 | The inactive state value of SPCK is logic level zero. |
| 1 | The inactive state value of SPCK is logic level one. |

**Bits 15:14 – CHMODE[1:0]** Channel Mode

| Value | Name | Description |
|---|---|---|
| 0 | NORMAL | Normal mode |

**Bit 8 – CPHA** SPI Clock Phase
CPHA determines which edge of SPCK causes data to change and which edge causes data to be captured. CPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.

Applicable if USART operates in SPI mode (USART_MODE = 0xE or 0xF):

| Value | Description |
|---|---|
| 0 | Data are changed on the leading edge of SPCK and captured on the following edge of SPCK. |
| 1 | Data are captured on the leading edge of SPCK and changed on the following edge of SPCK. |

**Bits 7:6 – CHRL[1:0]** Character Length

| Value | Name | Description |
|---|---|---|
| 3 | 8_BIT | Character length is 8 bits |

**Bits 5:4 – USCLKS[1:0]** Clock Selection

| Value | Name | Description |
|---|---|---|
| 0 | MCK | Peripheral clock is selected |
| 1 | DIV | Peripheral clock Divided (DIV= 8) is selected |
| 2 | GCLK | A PMC generic clock is selected |
| 3 | SCK | External pin SCK is selected |

**Bits 3:0 – USART_MODE[3:0]** USART Mode of Operation

| Value | Name | Description |
|---|---|---|
| 0xE | SPI_MASTER | SPI master |
| 0xF | SPI_SLAVE | SPI slave |

### 46.10.8 USART Interrupt Enable Register

**Name:** FLEX_US_IER
**Offset:** 0x208
**Reset:** –
**Property:** Write-only

For SPI-specific configurations, see USART Interrupt Enable Register (SPI_MODE).

For LIN-specific configurations, see USART Interrupt Enable Register (LIN_MODE).

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | MANE |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | CMP | | | CTSIC | | | |
| Access | | W | | | W | | | |
| Reset | | – | | | – | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | NACK | | | ITER | TXEMPTY | TIMEOUT |
| Access | | | W | | | W | W | W |
| Reset | | | – | | | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | RXBRK | TXRDY | RXRDY |
| Access | W | W | W | | | W | W | W |
| Reset | – | – | – | | | – | – | – |

**Bit 24 – MANE** Manchester Error Interrupt Enable

**Bit 22 – CMP** Comparison Interrupt Enable

**Bit 19 – CTSIC** Clear to Send Input Change Interrupt Enable

**Bit 13 – NACK** Non Acknowledge Interrupt Enable

**Bit 10 – ITER** Max number of Repetitions Reached Interrupt Enable

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Enable

**Bit 8 – TIMEOUT** Timeout Interrupt Enable

**Bit 7 – PARE** Parity Error Interrupt Enable

**Bit 6 – FRAME** Framing Error Interrupt Enable

**Bit 5 – OVRE** Overrun Error Interrupt Enable

**Bit 2 – RXBRK** Receiver Break Interrupt Enable

**Bit 1 – TXRDY** TXRDY Interrupt Enable

**Bit 0 – RXRDY** RXRDY Interrupt Enable

### 46.10.9 USART Interrupt Enable Register (SPI_MODE)

**Name:** FLEX_US_IER (SPI_MODE)
**Offset:** 0x208
**Reset:** –
**Property:** Write-only

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

This register can only be written if the WPITEN bit is cleared in the USART Write Protection Mode Register .

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | CMP | | | NSSE | | | |
| Access | | W | | | W | | | |
| Reset | | – | | | – | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | UNRE | TXEMPTY | |
| Access | | | | | | W | W | |
| Reset | | | | | | – | – | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | OVRE | | | | TXRDY | RXRDY |
| Access | | | W | | | | W | W |
| Reset | | | – | | | | – | – |

**Bit 22 – CMP** Comparison Interrupt Enable

**Bit 19 – NSSE** NSS Line (Driving CTS Pin) Rising or Falling Edge Event

**Bit 10 – UNRE** SPI Underrun Error Interrupt Enable

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Enable

**Bit 5 – OVRE** Overrun Error Interrupt Enable

**Bit 1 – TXRDY** TXRDY Interrupt Enable

**Bit 0 – RXRDY** RXRDY Interrupt Enable

### 46.10.10 USART Interrupt Enable Register (LIN_MODE)

**Name:** FLEX_US_IER (LIN_MODE)
**Offset:** 0x208
**Reset:** –
**Property:** Write-only

This configuration is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

This register can only be written if the WPITEN bit is cleared in the USART Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | LINHTE | LINSTE | LINSNRE | LINCE | LINIPE | LINISFE | LINBE | |
| Access | W | W | W | W | W | W | W | |
| Reset | – | – | – | – | – | – | – | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LINTC | LINID | LINBK | | | | TXEMPTY | TIMEOUT |
| Access | W | W | W | | | | W | W |
| Reset | – | – | – | | | | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| Access | W | W | W | | | | W | W |
| Reset | – | – | – | | | | – | – |

**Bit 31 – LINHTE** LIN Header Timeout Error Interrupt Enable

**Bit 30 – LINSTE** LIN Synch Tolerance Error Interrupt Enable

**Bit 29 – LINSNRE** LIN Slave Not Responding Error Interrupt Enable

**Bit 28 – LINCE** LIN Checksum Error Interrupt Enable

**Bit 27 – LINIPE** LIN Identifier Parity Interrupt Enable

**Bit 26 – LINISFE** LIN Inconsistent Synch Field Error Interrupt Enable

**Bit 25 – LINBE** LIN Bus Error Interrupt Enable

**Bit 15 – LINTC** LIN Transfer Completed Interrupt Enable

**Bit 14 – LINID** LIN Identifier Sent or LIN Identifier Received Interrupt Enable

**Bit 13 – LINBK** LIN Break Sent or LIN Break Received Interrupt Enable

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Enable

**Bit 8 – TIMEOUT** Timeout Interrupt Enable

**Bit 7 – PARE** Parity Error Interrupt Enable

**Bit 6 – FRAME** Framing Error Interrupt Enable

**Bit 5 – OVRE** Overrun Error Interrupt Enable

**Bit 1 – TXRDY** TXRDY Interrupt Enable

**Bit 0 – RXRDY** RXRDY Interrupt Enable

© 2018 Microchip Technology Inc.

### 46.10.11 USART Interrupt Disable Register

**Name:** FLEX_US_IDR
**Offset:** 0x20C
**Reset:** –
**Property:** Write-only

For SPI-specific configurations, see USART Interrupt Disable Register (SPI_MODE).

For LIN-specific configurations, see USART Interrupt Disable Register (LIN_MODE).

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | MANE |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | CMP | | | CTSIC | | | |
| Access | | W | | | W | | | |
| Reset | | – | | | – | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | NACK | | | ITER | TXEMPTY | TIMEOUT |
| Access | | | W | | | W | W | W |
| Reset | | | – | | | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | RXBRK | TXRDY | RXRDY |
| Access | W | W | W | | | W | W | W |
| Reset | – | – | – | | | – | – | – |

**Bit 24 – MANE** Manchester Error Interrupt Disable

**Bit 22 – CMP** Comparison Interrupt Disable

**Bit 19 – CTSIC** Clear to Send Input Change Interrupt Disable

**Bit 13 – NACK** Non Acknowledge Interrupt Disable

**Bit 10 – ITER** Max Number of Repetitions Reached Interrupt Disable

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Disable

**Bit 8 – TIMEOUT** Timeout Interrupt Disable

**Bit 7 – PARE** Parity Error Interrupt Disable

**Bit 6 – FRAME** Framing Error Interrupt Disable

**Bit 5 – OVRE** Overrun Error Interrupt Enable

**Bit 2 – RXBRK** Receiver Break Interrupt Disable

**Bit 1 – TXRDY** TXRDY Interrupt Disable

**Bit 0 – RXRDY** RXRDY Interrupt Disable

### 46.10.12 USART Interrupt Disable Register (SPI_MODE)

**Name:** FLEX_US_IDR (SPI_MODE)
**Offset:** 0x20C
**Reset:** –
**Property:** Write-only

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

This register can only be written if the WPITEN bit is cleared in the USART Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | CMP | | | NSSE | | | |
| Access | | W | | | W | | | |
| Reset | | – | | | – | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | UNRE | TXEMPTY | |
| Access | | | | | | W | W | |
| Reset | | | | | | – | – | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | OVRE | | | | TXRDY | RXRDY |
| Access | | | W | | | | W | W |
| Reset | | | – | | | | – | – |

**Bit 22 – CMP** Comparison Interrupt Disable

**Bit 19 – NSSE** NSS Line (Driving CTS Pin) Rising or Falling Edge Event

**Bit 10 – UNRE** SPI Underrun Error Interrupt Disable

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Disable

**Bit 5 – OVRE** Overrun Error Interrupt Disable

**Bit 1 – TXRDY** TXRDY Interrupt Disable

**Bit 0 – RXRDY** RXRDY Interrupt Disable

### 46.10.13 USART Interrupt Disable Register (LIN_MODE)

**Name:** FLEX_US_IDR (LIN_MODE)
**Offset:** 0x20C
**Reset:** –
**Property:** Write-only

This configuration is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

This register can only be written if the WPITEN bit is cleared in the USART Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | LINHTE | LINSTE | LINSNRE | LINCE | LINIPE | LINISFE | LINBE | |
| Access | W | W | W | W | W | W | W | |
| Reset | – | – | – | – | – | – | – | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LINTC | LINID | LINBK | | | | TXEMPTY | TIMEOUT |
| Access | W | W | W | | | | W | W |
| Reset | – | – | – | | | | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| Access | W | W | W | | | | W | W |
| Reset | – | – | – | | | | – | – |

**Bit 31 – LINHTE** LIN Header Timeout Error Interrupt Disable

**Bit 30 – LINSTE** LIN Synch Tolerance Error Interrupt Disable

**Bit 29 – LINSNRE** LIN Slave Not Responding Error Interrupt Disable

**Bit 28 – LINCE** LIN Checksum Error Interrupt Disable

**Bit 27 – LINIPE** LIN Identifier Parity Interrupt Disable

**Bit 26 – LINISFE** LIN Inconsistent Synch Field Error Interrupt Disable

**Bit 25 – LINBE** LIN Bus Error Interrupt Disable

**Bit 15 – LINTC** LIN Transfer Completed Interrupt Disable

**Bit 14 – LINID** LIN Identifier Sent or LIN Identifier Received Interrupt Disable

**Bit 13 – LINBK** LIN Break Sent or LIN Break Received Interrupt Disable

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Disable

**Bit 8 – TIMEOUT** Timeout Interrupt Disable

**Bit 7 – PARE** Parity Error Interrupt Disable

**Bit 6 – FRAME** Framing Error Interrupt Disable

**Bit 5 – OVRE** Overrun Error Interrupt Disable

**Bit 1 – TXRDY** TXRDY Interrupt Disable

**Bit 0 – RXRDY** RXRDY Interrupt Disable

### 46.10.14 USART Interrupt Mask Register

**Name:** FLEX_US_IMR
**Offset:** 0x210
**Reset:** 0x00000000
**Property:** Read-only

For SPI-specific configurations, see USART Interrupt Mask Register (SPI_MODE).

For LIN-specific configurations, see USART Interrupt Mask Register (LIN_MODE).

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | MANE |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | CMP | | | CTSIC | | | |
| Access | | R | | | R | | | |
| Reset | | 0 | | | 0 | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | NACK | | | ITER | TXEMPTY | TIMEOUT |
| Access | | | R | | | R | R | R |
| Reset | | | 0 | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | RXBRK | TXRDY | RXRDY |
| Access | R | R | R | | | R | R | R |
| Reset | 0 | 0 | 0 | | | 0 | 0 | 0 |

**Bit 24 – MANE** Manchester Error Interrupt Mask

**Bit 22 – CMP** Comparison Interrupt Mask

**Bit 19 – CTSIC** Clear to Send Input Change Interrupt Mask

**Bit 13 – NACK** Non Acknowledge Interrupt Mask

**Bit 10 – ITER** Max Number of Repetitions Reached Interrupt Mask

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Mask

**Bit 8 – TIMEOUT** Timeout Interrupt Mask

**Bit 7 – PARE** Parity Error Interrupt Mask

**Bit 6 – FRAME** Framing Error Interrupt Mask

**Bit 5 – OVRE** Overrun Error Interrupt Mask

**Bit 2 – RXBRK** Receiver Break Interrupt Mask

**Bit 1 – TXRDY** TXRDY Interrupt Mask

**Bit 0 – RXRDY** RXRDY Interrupt Mask

### 46.10.15 USART Interrupt Mask Register (SPI_MODE)

**Name:** FLEX_US_IMR (SPI_MODE)
**Offset:** 0x210
**Reset:** 0x00000000
**Property:** Read-only

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | CMP | | | NSSE | | | |
| Access | | R | | | R | | | |
| Reset | | 0 | | | 0 | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | UNRE | TXEMPTY | |
| Access | | | | | | R | R | |
| Reset | | | | | | 0 | 0 | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | OVRE | | | | TXRDY | RXRDY |
| Access | | | R | | | | R | R |
| Reset | | | 0 | | | | 0 | 0 |

**Bit 22 – CMP** Comparison Interrupt Mask

**Bit 19 – NSSE** NSS Line (Driving CTS Pin) Rising or Falling Edge Event

**Bit 10 – UNRE** SPI Underrun Error Interrupt Mask

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Mask

**Bit 5 – OVRE** Overrun Error Interrupt Mask

**Bit 1 – TXRDY** TXRDY Interrupt Mask

**Bit 0 – RXRDY** RXRDY Interrupt Mask

### 46.10.16 USART Interrupt Mask Register (LIN_MODE)

**Name:** FLEX_US_IMR (LIN_MODE)
**Offset:** 0x210
**Reset:** 0x00000000
**Property:** Read-only

This configuration is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | LINHTE | LINSTE | LINSNRE | LINCE | LINIPE | LINISFE | LINBE | |
| Access | R | R | R | R | R | R | R | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LINTC | LINID | LINBK | | | | TXEMPTY | TIMEOUT |
| Access | R | R | R | | | | R | R |
| Reset | 0 | 0 | 0 | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| Access | R | R | R | | | | R | R |
| Reset | 0 | 0 | 0 | | | | 0 | 0 |

**Bit 31 – LINHTE** LIN Header Timeout Error Interrupt Mask

**Bit 30 – LINSTE** LIN Synch Tolerance Error Interrupt Mask

**Bit 29 – LINSNRE** LIN Slave Not Responding Error Interrupt Mask

**Bit 28 – LINCE** LIN Checksum Error Interrupt Mask

**Bit 27 – LINIPE** LIN Identifier Parity Interrupt Mask

**Bit 26 – LINISFE** LIN Inconsistent Synch Field Error Interrupt Mask

**Bit 25 – LINBE** LIN Bus Error Interrupt Mask

**Bit 15 – LINTC** LIN Transfer Completed Interrupt Mask

**Bit 14 – LINID** LIN Identifier Sent or LIN Identifier Received Interrupt Mask

**Bit 13 – LINBK** LIN Break Sent or LIN Break Received Interrupt Mask

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Mask

**Bit 8 – TIMEOUT** Timeout Interrupt Mask

**Bit 7 – PARE** Parity Error Interrupt Mask

**Bit 6 – FRAME** Framing Error Interrupt Mask

**Bit 5 – OVRE** Overrun Error Interrupt Mask

**Bit 1 – TXRDY** TXRDY Interrupt Mask

**Bit 0 – RXRDY** RXRDY Interrupt Mask

### 46.10.17 USART Channel Status Register

**Name:** FLEX_US_CSR
**Offset:** 0x214
**Reset:** 0x00000000
**Property:** Read-only

For SPI-specific configurations, see USART Channel Status Register (SPI_MODE).

For LIN-specific configurations, see USART Channel Status Register (LIN_MODE).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | MANE |
| Access | | | | | | | | R |
| Reset | | | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CTS | CMP | | | CTSIC | | | |
| Access | R | R | | | R | | | |
| Reset | – | – | | | – | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | NACK | | | ITER | TXEMPTY | TIMEOUT |
| Access | | | R | | | R | R | R |
| Reset | | | – | | | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | RXBRK | TXRDY | RXRDY |
| Access | R | R | R | | | R | R | R |
| Reset | – | – | – | | | – | – | – |

**Bit 24 – MANE** Manchester Error

| Value | Description |
|---|---|
| 0 | No Manchester error has been detected since the last RSTSTA command was issued. |
| 1 | At least one Manchester error has been detected since the last RSTSTA command was issued. |

**Bit 23 – CTS** Image of CTS Input

| Value | Description |
|---|---|
| 0 | CTS input is driven low. |
| 1 | CTS input is driven high. |

**Bit 22 – CMP** Comparison Status

| Value | Description |
|---|---|
| 0 | No received character matched the comparison criteria programmed in VAL1, VAL2 fields and CMPPAR bit in since the last RSTSTA command was issued. |
| 1 | A received character matched the comparison criteria since the last RSTSTA command was issued. |

**Bit 19 – CTSIC** Clear to Send Input Change Flag

| Value | Description |
|---|---|
| 0 | No input change has been detected on the CTS pin since the last read of FLEX_US_CSR. |
| 1 | At least one input change has been detected on the CTS pin since the last read of FLEX_US_CSR. |

**Bit 13 – NACK** Non Acknowledge Interrupt

| Value | Description |
|---|---|
| 0 | Non acknowledge has not been detected since the last RSTNACK. |
| 1 | At least one non acknowledge has been detected since the last RSTNACK. |

**Bit 10 – ITER** Max Number of Repetitions Reached

| Value | Description |
|---|---|
| 0 | Maximum number of repetitions has not been reached since the last RSTIT command was issued. |
| 1 | Maximum number of repetitions has been reached since the last RSTIT command was issued. |

**Bit 9 – TXEMPTY** Transmitter Empty (cleared by writing FLEX_US_THR)

| Value | Description |
|---|---|
| 0 | There are characters in either FLEX_US_THR or the Transmit Shift Register, or the transmitter is disabled. |
| 1 | There are no characters in FLEX_US_THR, nor in the Transmit Shift Register. |

**Bit 8 – TIMEOUT** Receiver Timeout

| Value | Description |
|---|---|
| 0 | There has not been a timeout since the last Start Timeout command (FLEX_US_CR.STTTO) or the Timeout Register is 0. |
| 1 | There has been a timeout since the last Start Timeout command (FLEX_US_CR.STTTO). |

**Bit 7 – PARE** Parity Error

| Value | Description |
|---|---|
| 0 | No parity error has been detected since the last RSTSTA command was issued. |
| 1 | At least one parity error has been detected since the last RSTSTA command was issued. |

**Bit 6 – FRAME** Framing Error

| Value | Description |
|---|---|
| 0 | No stop bit has been detected low since the last RSTSTA command was issued. |
| 1 | At least one stop bit has been detected low since the last RSTSTA command was issued. |

**Bit 5 – OVRE** Overrun Error

| Value | Description |
|---|---|
| 0 | No overrun error has occurred since the last RSTSTA command was issued. |
| 1 | At least one overrun error has occurred since the last RSTSTA command was issued. |

**Bit 2 – RXBRK** Break Received/End of Break

| Value | Description |
|---|---|
| 0 | No break received or end of break detected since the last RSTSTA command was issued. |
| 1 | Break received or end of break detected since the last RSTSTA command was issued. |

**Bit 1 – TXRDY** Transmitter Ready (cleared by writing FLEX_US_THR)
When FIFOs are disabled:

0: A character in FLEX_US_THR is waiting to be transferred to the Transmit Shift Register, or an STTBRK command has been requested, or the transmitter is disabled. As soon as the transmitter is enabled, TXRDY becomes 1.

1: There is no character in FLEX_US_THR.

When FIFOs are enabled:

0: Transmit FIFO is full and cannot accept more data.

1: Transmit FIFO is not full; one or more data can be written according to TXRDYM field configuration.

TXRDY behavior with FIFO enabled is illustrated in 46.7.11.5 TXEMPTY, TXRDY and RXRDY Behavior.

**Bit 0 – RXRDY** Receiver Ready (cleared by reading FLEX_US_RHR)
When FIFOs are disabled:

0: No complete character has been received since the last read of FLEX_US_RHR or the receiver is disabled. If characters were received when the receiver was disabled, RXRDY changes to 1 when the receiver is enabled.

1: At least one complete character has been received and FLEX_US_RHR has not yet been read.

When FIFOs are enabled:

0: Receive FIFO is empty; no data to read

1: At least one unread data is in the Receive FIFO

RXRDY behavior with FIFO enabled is illustrated in 46.7.11.5 TXEMPTY, TXRDY and RXRDY Behavior.

### 46.10.18 USART Channel Status Register (SPI_MODE)

**Name:** FLEX_US_CSR (SPI_MODE)
**Offset:** 0x214
**Reset:** –
**Property:** Read-only

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NSS | CMP | | | NSSE | | | |
| Access | R | R | | | R | | | |
| Reset | – | – | | | – | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | UNRE | TXEMPTY | |
| Access | | | | | | R | R | |
| Reset | | | | | | – | – | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | OVRE | | | | TXRDY | RXRDY |
| Access | | | R | | | | R | R |
| Reset | | | – | | | | – | – |

**Bit 23 – NSS** Image of NSS Line

| Value | Description |
|---|---|
| 0 | NSS line is driven low (if NSSE = 1, falling edge occurred on NSS line). |
| 1 | NSS line is driven high (if NSSE = 1, rising edge occurred on NSS line). |

**Bit 22 – CMP** Comparison Match

| Value | Description |
|---|---|
| 0 | No received character matched the comparison criteria programmed in VAL1, VAL2 fields and CMPPAR bit in FLEX_US_CMPR since the last RSTSTA command was issued. |
| 1 | A received character matched the comparison criteria since the last RSTSTA command was issued. |

**Bit 19 – NSSE** NSS Line (Driving CTS Pin) Rising or Falling Edge Event (cleared on read)

| Value | Description |
|---|---|
| 0 | No NSS line event has been detected since the last read of FLEX_US_CSR. |
| 1 | A rising or falling edge has been detected on the NSS line since the last read of FLEX_US_CSR. |

**Bit 10 – UNRE** Underrun Error

| Value | Description |
|---|---|
| 0 | No SPI underrun error has occurred since the last RSTSTA command was issued. |
| 1 | At least one SPI underrun error has occurred since the last RSTSTA command was issued. |

**Bit 9 – TXEMPTY** Transmitter Empty (cleared by writing FLEX_US_THR)

| Value | Description |
|---|---|
| 0 | There are characters in either FLEX_US_THR or the Transmit Shift Register, or the transmitter is disabled. |
| 1 | There are no characters in FLEX_US_THR, nor in the Transmit Shift Register. |

**Bit 5 – OVRE** Overrun Error

| Value | Description |
|---|---|
| 0 | No overrun error has occurred since the last RSTSTA command was issued. |
| 1 | At least one overrun error has occurred since the last RSTSTA command was issued. |

**Bit 1 – TXRDY** Transmitter Ready (cleared by writing FLEX_US_THR)

| Value | Description |
|---|---|
| 0 | A character in FLEX_US_THR is waiting to be transferred to the Transmit Shift Register, or the transmitter is disabled. As soon as the transmitter is enabled, TXRDY becomes 1. |
| 1 | There is no character in FLEX_US_THR. |

**Bit 0 – RXRDY** Receiver Ready (cleared by reading FLEX_US_RHR)

| Value | Description |
|---|---|
| 0 | No complete character has been received since the last read of FLEX_US_RHR or the receiver is disabled. If characters were being received when the receiver was disabled, RXRDY changes to 1 when the receiver is enabled. |
| 1 | At least one complete character has been received and FLEX_US_RHR has not yet been read. |

### 46.10.19 USART Channel Status Register (LIN_MODE)

**Name:** FLEX_US_CSR (LIN_MODE)
**Offset:** 0x214
**Reset:** –
**Property:** Read-only

This configuration is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | LINHTE | LINSTE | LINSNRE | LINCE | LINIPE | LINISFE | LINBE | |
| Access | R | R | R | R | R | R | R | |
| Reset | – | – | – | – | – | – | – | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | LINBLS | | | | | | | |
| Access | R | | | | | | | |
| Reset | – | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LINTC | LINID | LINBK | | | | TXEMPTY | TIMEOUT |
| Access | R | R | R | | | | R | R |
| Reset | – | – | – | | | | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| Access | R | R | R | | | | R | R |
| Reset | – | – | – | | | | – | – |

**Bit 31 – LINHTE** LIN Header Timeout Error

| Value | Description |
|---|---|
| 0 | No LIN header timeout error has been detected since the last RSTSTA command was issued. |
| 1 | A LIN header timeout error has been detected since the last RSTSTA command was issued. |

**Bit 30 – LINSTE** LIN Synch Tolerance Error

| Value | Description |
|---|---|
| 0 | No LIN synch tolerance error has been detected since the last RSTSTA command was issued. |
| 1 | A LIN synch tolerance error has been detected since the last RSTSTA command was issued. |

**Bit 29 – LINSNRE** LIN Slave Not Responding Error

| Value | Description |
|---|---|
| 0 | No LIN slave not responding error has been detected since the last RSTSTA command was issued. |

| Value | Description |
|---|---|
| 1 | A LIN slave not responding error has been detected since the last RSTSTA command was issued. |

**Bit 28 – LINCE** LIN Checksum Error

| Value | Description |
|---|---|
| 0 | No LIN checksum error has been detected since the last RSTSTA command was issued. |
| 1 | A LIN checksum error has been detected since the last RSTSTA command was issued. |

**Bit 27 – LINIPE** LIN Identifier Parity Error

| Value | Description |
|---|---|
| 0 | No LIN identifier parity error has been detected since the last RSTSTA command was issued. |
| 1 | A LIN identifier parity error has been detected since the last RSTSTA command was issued. |

**Bit 26 – LINISFE** LIN Inconsistent Synch Field Error

| Value | Description |
|---|---|
| 0 | No LIN inconsistent synch field error has been detected since the last RSTSTA |
| 1 | The USART is configured as a slave node and a LIN Inconsistent synch field error has been detected since the last RSTSTA command was issued. |

**Bit 25 – LINBE** LIN Bit Error

| Value | Description |
|---|---|
| 0 | No bit error has been detected since the last RSTSTA command was issued. |
| 1 | A bit error has been detected since the last RSTSTA command was issued. |

**Bit 23 – LINBLS** LIN Bus Line Status

| Value | Description |
|---|---|
| 0 | LIN bus line is set to 0. |
| 1 | LIN bus line is set to 1. |

**Bit 15 – LINTC** LIN Transfer Completed

| Value | Description |
|---|---|
| 0 | The USART is idle or a LIN transfer is ongoing. |
| 1 | A LIN transfer has been completed since the last RSTSTA command was issued. |

**Bit 14 – LINID** LIN Identifier Sent or LIN Identifier Received
If USART operates in LIN Master mode (USART_MODE = 0xA):

0: No LIN identifier has been sent since the last RSTSTA command was issued.

1: At least one LIN identifier has been sent since the last RSTSTA command was issued.

If USART operates in LIN Slave mode (USART_MODE = 0xB):

0: No LIN identifier has been received since the last RSTSTA command was issued.

1: At least one LIN identifier has been received since the last RSTSTA.

**Bit 13 – LINBK** LIN Break Sent or LIN Break Received
Applicable if USART operates in LIN Master mode (USART_MODE = 0xA):

0: No LIN break has been sent since the last RSTSTA command was issued.

1: At least one LIN break has been sent since the last RSTSTA.

If USART operates in LIN Slave mode (USART_MODE = 0xB):

0: No LIN break has received sent since the last RSTSTA command was issued.

1: At least one LIN break has been received since the last RSTSTA command was issued.

**Bit 9 – TXEMPTY** Transmitter Empty (cleared by writing FLEX_US_THR)

| Value | Description |
|---|---|
| 0 | There are characters in either FLEX_US_THR or the Transmit Shift Register, or the transmitter is disabled. |
| 1 | There are no characters in FLEX_US_THR, nor in the Transmit Shift Register. |

**Bit 8 – TIMEOUT** Receiver Timeout

| Value | Description |
|---|---|
| 0 | There has not been a timeout since the last start timeout command (FLEX_US_CR.STTTO) or the Timeout Register is 0. |
| 1 | There has been a timeout since the last start timeout command (FLEX_US_CR.STTTO). |

**Bit 7 – PARE** Parity Error

| Value | Description |
|---|---|
| 0 | No parity error has been detected since the last RSTSTA command was issued. |
| 1 | At least one parity error has been detected since the last RSTSTA command was issued. |

**Bit 6 – FRAME** Framing Error

| Value | Description |
|---|---|
| 0 | No stop bit has been detected low since the last RSTSTA command was issued. |
| 1 | At least one stop bit has been detected low since the last RSTSTA command was issued. |

**Bit 5 – OVRE** Overrun Error

| Value | Description |
|---|---|
| 0 | No overrun error has occurred since the last RSTSTA command was issued. |
| 1 | At least one overrun error has occurred since the last RSTSTA command was issued. |

**Bit 1 – TXRDY** Transmitter Ready (cleared by writing FLEX_US_THR)

| Value | Description |
|---|---|
| 0 | A character in FLEX_US_THR is waiting to be transferred to the Transmit Shift Register, or the transmitter is disabled. As soon as the transmitter is enabled, TXRDY becomes 1. |
| 1 | There is no character in FLEX_US_THR. |

**Bit 0 – RXRDY** Receiver Ready (cleared by reading FLEX_US_RHR)

| Value | Description |
|---|---|
| 0 | No complete character has been received since the last read of FLEX_US_RHR or the receiver is disabled. If characters were being received when the receiver was disabled, RXRDY changes to 1 when the receiver is enabled. |
| 1 | At least one complete character has been received and FLEX_US_RHR has not yet been read. |

### 46.10.20 USART Receive Holding Register

**Name:** FLEX_US_RHR
**Offset:** 0x218
**Reset:** 0x00000000
**Property:** Read-only

If FIFO is enabled (FLEX_US_CR.FIFOEN bit) and FLEX_US_FMR.RXRDYM = 0, see 46.7.11.6 USART Single Data Mode for details.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RXSYNH | | | | | | | RXCHR[8:8] |
| Access | R | | | | | | | R |
| Reset | 0 | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXCHR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 15 – RXSYNH** Received Sync

| Value | Description |
|---|---|
| 0 | Last character received is a data. |
| 1 | Last character received is a command. |

**Bits 8:0 – RXCHR[8:0]** Received Character
Last character received if RXRDY is set.

### 46.10.21 USART Receive Holding Register (FIFO Multi Data)

**Name:** FLEX_US_RHR (FIFO_MULTI_DATA)
**Offset:** 0x218
**Reset:** 0x00000000
**Property:** Read-only

If FIFO is enabled (FLEX_US_CR.FIFOEN bit) and FLEX_US_FMR.RXRDYM > 0, see 46.7.11.7 USART Multiple Data Mode for details.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RXCHR3[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RXCHR2[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RXCHR1[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXCHR0[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0:7, 8:15, 16:23, 24:31 – RXCHRx** Received Character
First unread character in the Receive FIFO if RXRDY is set.

### 46.10.22 USART Transmit Holding Register

**Name:** FLEX_US_THR
**Offset:** 0x21C
**Reset:** –
**Property:** Write-only

If FIFO is enabled (FLEX_US_CR.FIFOEN bit) and FLEX_US_FMR.TXRDY = 0, see 46.7.11.6 USART Single Data Mode for details.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TXSYNH | | | | | | | TXCHR[8:8] |
| Access | W | | | | | | | W |
| Reset | – | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXCHR[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 15 – TXSYNH** Sync Field to be Transmitted

| Value | Description |
|---|---|
| 0 | The next character sent is encoded as a data. Start frame delimiter is DATA SYNC. |
| 1 | The next character sent is encoded as a command. Start frame delimiter is COMMAND SYNC. |

**Bits 8:0 – TXCHR[8:0]** Character to be Transmitted
The next character to be transmitted after the current character if TXRDY is not set.

### 46.10.23 USART Transmit Holding Register (FIFO Multi Data)

**Name:** FLEX_US_THR (FIFO_MULTI_DATA)
**Offset:** 0x21C
**Reset:** –
**Property:** Write-only

If FIFO is enabled (FLEX_US_CR.FIFOEN bit) and FLEX_US_FMR.TXRDY > 0, see 46.7.11.7 USART Multiple Data Mode for details.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TXCHR3[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TXCHR2[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TXCHR1[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXCHR0[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0:7, 8:15, 16:23, 24:31 – TXCHRx** Character to be Transmitted
Next character to be transmitted.

### 46.10.24 USART Baud Rate Generator Register

**Name:** FLEX_US_BRGR
**Offset:** 0x220
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |
| | | | | | | FP[2:0] | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | CD[15:8] | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | CD[7:0] | | | | | | | |

**Bits 18:16 – FP[2:0]** Fractional Part

**WARNING** When the value of field FP is greater than 0, the SCK (oversampling clock) generates nonconstant duty cycles. The SCK high duration is increased by “selected clock” period from time to time. The duty cycle depends on the value of the CD field.

| Value | Description |
|---|---|
| 0 | Fractional divider is disabled. |
| 1–7 | Baud rate resolution, defined by FP × 1/8. |

**Bits 15:0 – CD[15:0]** Clock Divider

| CD | USART_MODE ≠ ISO7816 | | | USART_MODE = ISO7816 |
|---|---|---|---|---|
| | SYNC = 0 | | SYNC = 1 or USART_MODE = SPI (master or Slave) | |
| | OVER = 0 | OVER = 1 | | |

| 0 | Baud Rate Clock Disabled | | | |
|---|---|---|---|---|
| 1 to 65535 | CD = Selected Clock / (16 × Baud Rate) | CD = Selected Clock / (8 × Baud Rate) | CD = Selected Clock / Baud Rate | CD = Selected Clock / (FI_DI_RATIO × Baud Rate) |

### 46.10.25 USART Receiver Timeout Register

**Name:** FLEX_US_RTOR
**Offset:** 0x224
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | TO[16:16] |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TO[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TO[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 16:0 – TO[16:0]** Timeout Value

| Value | Description |
|---|---|
| 0 | The receiver timeout is disabled. |
| 1–131071 | The receiver timeout is enabled and the timeout delay is TO × bit period. |

### 46.10.26 USART Transmitter Timeguard Register

**Name:** FLEX_US_TTGR
**Offset:** 0x228
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TG[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – TG[7:0]** Timeguard Value

| Value | Description |
|---|---|
| 0 | The transmitter timeguard is disabled. |
| 1–255 | The transmitter timeguard is enabled and TG is timeguard delay / bit period. |

### 46.10.27 USART FI DI RATIO Register

**Name:** FLEX_US_FIDI
**Offset:** 0x240
**Reset:** 0x174
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FI_DI_RATIO[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FI_DI_RATIO[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |

**Bits 15:0 – FI_DI_RATIO[15:0]** FI Over DI Ratio Value

| Value | Description |
|---|---|
| 0 | If ISO7816 mode is selected, the baud rate generator generates no signal. |
| 1–2 | Do not use. |
| 3–65535 | If ISO7816 mode is selected, the baud rate is the clock provided on SCK divided by FI_DI_RATIO. |

### 46.10.28 USART Number of Errors Register

**Name:** FLEX_US_NER
**Offset:** 0x244
**Reset:** –
**Property:** Read-only

This register is relevant only if USART_MODE = 0x4 or 0x6 in the USART Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NB_ERRORS[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | – | – | – | – | – | – | – | – |

**Bits 7:0 – NB_ERRORS[7:0]** Number of Errors
Total number of errors that occurred during an ISO7816 transfer. This register automatically clears when read.

### 46.10.29 USART IrDA FILTER Register

**Name:** FLEX_US_IF
**Offset:** 0x24C
**Reset:** 0x00000000
**Property:** Read/Write

This register is relevant only if USART_MODE = 0x8 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IRDA_FILTER[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – IRDA_FILTER[7:0]** IrDA Filter
The IRDA_FILTER value must be defined to meet the following criteria:

$t_{peripheral\ clock} \times (IRDA_FILTER + 3) < 1.41\ \mu s$

### 46.10.30 USART Manchester Configuration Register

**Name:** FLEX_US_MAN
**Offset:** 0x250
**Reset:** 0xB0011004
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RXIDLEV | DRIFT | ONE | RX_MPOL | | | RX_PP[1:0] | |
| Access | R/W | R/W | R/W | R/W | | | R/W | R/W |
| Reset | 1 | 0 | 1 | 1 | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RX_PL[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | TX_MPOL | | | TX_PP[1:0] | |
| Access | | | | R/W | | | R/W | R/W |
| Reset | | | | 1 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TX_PL[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 1 | 0 | 0 |

**Bit 31 – RXIDLEV** Receiver Idle Value

| Value | Description |
|---|---|
| 0 | Receiver line idle value is 0. |
| 1 | Receiver line idle value is 1. |

**Bit 30 – DRIFT** Drift Compensation

| Value | Description |
|---|---|
| 0 | The USART cannot recover from an important clock drift. |
| 1 | The USART can recover from clock drift. The 16X Clock mode must be enabled. |

**Bit 29 – ONE** Must Be Set to 1
Bit 29 must always be set to 1 when programming the FLEX_US_MAN register.

**Bit 28 – RX_MPOL** Receiver Manchester Polarity

| Value | Description |
|---|---|
| 0 | Logic zero is coded as a zero-to-one transition, Logic one is coded as a one-to-zero transition. |

| Value | Description |
|---|---|
| 1 | Logic zero is coded as a one-to-zero transition, Logic one is coded as a zero-to-one transition. |

**Bits 25:24 – RX_PP[1:0]** Receiver Preamble Pattern detected
The following values assume that RX_MPOL field is not set:

| Value | Name | Description |
|---|---|---|
| 0 | ALL_ONE | The preamble is composed of ‘1’s. |
| 1 | ALL_ZERO | The preamble is composed of ‘0’s. |
| 2 | ZERO_ONE | The preamble is composed of ‘01’s. |
| 3 | ONE_ZERO | The preamble is composed of ‘10’s. |

**Bits 19:16 – RX_PL[3:0]** Receiver Preamble Length

| Value | Description |
|---|---|
| 0 | The receiver preamble pattern detection is disabled. |
| 1–15 | The detected preamble length is RX_PL × Bit Period. |

**Bit 12 – TX_MPOL** Transmitter Manchester Polarity

| Value | Description |
|---|---|
| 0 | Logic zero is coded as a zero-to-one transition, Logic one is coded as a one-to-zero transition. |
| 1 | Logic zero is coded as a one-to-zero transition, Logic one is coded as a zero-to-one transition. |

**Bits 9:8 – TX_PP[1:0]** Transmitter Preamble Pattern
The following values assume that TX_MPOL field is not set:

| Value | Name | Description |
|---|---|---|
| 0 | ALL_ONE | The preamble is composed of ‘1’s. |
| 1 | ALL_ZERO | The preamble is composed of ‘0’s. |
| 2 | ZERO_ONE | The preamble is composed of ‘01’s. |
| 3 | ONE_ZERO | The preamble is composed of ‘10’s. |

**Bits 3:0 – TX_PL[3:0]** Transmitter Preamble Length

| Value | Description |
|---|---|
| 0 | The transmitter preamble pattern generation is disabled. |
| 1–15 | The preamble length is TX_PL × Bit Period. |

### 46.10.31 USART LIN Mode Register

**Name:** FLEX_US_LINMR
**Offset:** 0x254
**Reset:** 0x00000000
**Property:** Read/Write

This register is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | SYNCDIS | PDCM |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DLC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WKUPTYP | FSDIS | DLM | CHKTYP | CHKDIS | PARDIS | NACT[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 17 – SYNCDIS** Synchronization Disable

| Value | Description |
|---|---|
| 0 | The synchronization procedure is performed in LIN slave node configuration. |
| 1 | The synchronization procedure is not performed in LIN slave node configuration. |

**Bit 16 – PDCM** DMAC Mode

| Value | Description |
|---|---|
| 0 | The LIN mode register FLEX_US_LINMR is not written by the DMAC. |
| 1 | The LIN mode register FLEX_US_LINMR (excepting that flag) is written by the DMAC. |

**Bits 15:8 – DLC[7:0]** Data Length Control

| Value | Description |
|---|---|
| 0-255 | Defines the response data length if DLM = 0, in that case the response data length is equal to DLC+1 bytes. |

**Bit 7 – WKUPTYP** Wake-up Signal Type

| Value | Description |
|---|---|
| 0 | Setting the LINWKUP bit in the control register sends a LIN 2.0 wake-up signal. |

| Value | Description |
|---|---|
| 1 | Setting the LINWKUP bit in the control register sends a LIN 1.3 wake-up signal. |

**Bit 6 – FSDIS** Frame Slot Mode Disable

| Value | Description |
|---|---|
| 0 | The Frame Slot mode is enabled. |
| 1 | The Frame Slot mode is disabled. |

**Bit 5 – DLM** Data Length Mode

| Value | Description |
|---|---|
| 0 | The response data length is defined by the DLC field of this register. |
| 1 | The response data length is defined by the bits 5 and 6 of the identifier (FLEX_US_LINIR.IDCHR). |

**Bit 4 – CHKTYP** Checksum Type

| Value | Description |
|---|---|
| 0 | LIN 2.0 “enhanced” checksum |
| 1 | LIN 1.3 “classic” checksum |

**Bit 3 – CHKDIS** Checksum Disable

| Value | Description |
|---|---|
| 0 | In master node configuration, the checksum is computed and sent automatically. In slave node configuration, the checksum is checked automatically. |
| 1 | Whatever the node configuration is, the checksum is not computed/sent and it is not checked. |

**Bit 2 – PARDIS** Parity Disable

| Value | Description |
|---|---|
| 0 | In master node configuration, the identifier parity is computed and sent automatically. In master node and slave node configuration, the parity is checked automatically. |
| 1 | Whatever the node configuration is, the Identifier parity is not computed/sent and it is not checked. |

**Bits 1:0 – NACT[1:0]** LIN Node Action
Values which are not listed in the table must be considered as “reserved”.

| Value | Name | Description |
|---|---|---|
| 0 | PUBLISH | The USART transmits the response. |
| 1 | SUBSCRIBE | The USART receives the response. |
| 2 | IGNORE | The USART does not transmit and does not receive the response. |

### 46.10.32 USART LIN Identifier Register

**Name:** FLEX_US_LINIR
**Offset:** 0x258
**Reset:** 0x00000000
**Property:** Read/Write

Write is possible only in LIN master node configuration.

This register is relevant only if USART_MODE = 0xA or 0xB in USART Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IDCHR[7:0] | | | | | | | |
| Access | R/W-R | R/W-R | R/W-R | R/W-R | R/W-R | R/W-R | R/W-R | R/W-R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – IDCHR[7:0]** Identifier Character
If USART_MODE = 0xA (master node configuration):

- IDCHR is Read/Write and its value is the identifier character to be transmitted.

If USART_MODE = 0xB (slave node configuration):

- IDCHR is Read-only and its value is the last identifier character that has been received.

### 46.10.33 USART LIN Baud Rate Register

**Name:** FLEX_US_LINBRR
**Offset:** 0x25C
**Reset:** 0x00000000
**Property:** Read-only

This register is relevant only if USART_MODE = 0xA or 0xB in USART Mode Register.

Returns the baud rate value after the synchronization process completion.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | LINFP[2:0] | | |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LINCD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LINCD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 18:16 – LINFP[2:0]** Fractional Part after Synchronization

**Bits 15:0 – LINCD[15:0]** Clock Divider after Synchronization

### 46.10.34 USART Comparison Register

**Name:** FLEX_US_CMPR
**Offset:** 0x290
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | VAL2[8:8] |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | VAL2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | CMPPAR | | CMPMODE | | | | VAL1[8:8] |
| Access | | R/W | | R/W | | | | R/W |
| Reset | | 0 | | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | VAL1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 24:16 – VAL2[8:0]** Second Comparison Value for Received Character

| Value | Description |
|---|---|
| 0-511 | The received character must be lower than or equal to the value of VAL2 and higher than or equal to VAL1 to set the FLEX_US_CSR.CMP flag. |

**Bit 14 – CMPPAR** Compare Parity

| Value | Description |
|---|---|
| 0 | The parity is not checked and a bad parity cannot prevent from waking up the system. |
| 1 | The parity is checked and a matching condition on data can be cancelled by an error on parity bit, so no wakeup is performed. |

**Bit 12 – CMPMODE** Comparison Mode

| Value | Name | Description |
|---|---|---|
| 0 | FLAG_ONLY | Any character is received and comparison function drives CMP flag. |
| 1 | START_CONDITION | Comparison condition must be met to start reception. |
| 2 | FILTER | Comparison must be met to receive the current data only |

**Bits 8:0 – VAL1[8:0]** First Comparison Value for Received Character

| Value | Description |
|---|---|
| 0–511 | The received character must be higher than or equal to the value of VAL1 and lower than or equal to VAL2 to set the FLEX_US_CSR.CMP flag. |

### 46.10.35 USART FIFO Mode Register

**Name:** FLEX_US_FMR
**Offset:** 0x2A0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | RXFTHRES2[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | RXFTHRES[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | TXFTHRES[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FRTSC | | RXRDYM[1:0] | | | | TXRDYM[1:0] | |
| Access | R/W | | R/W | R/W | | | R/W | R/W |
| Reset | 0 | | 0 | 0 | | | 0 | 0 |

**Bits 29:24 – RXFTHRES2[5:0]** Receive FIFO Threshold 2

| Value | Description |
|---|---|
| 0–32 | Defines the Receive FIFO threshold 2 value (number of data). The FLEX_US_FESR.RXFTHF2 flag will be set when Receive FIFO goes from “above” threshold state to “equal to or below” threshold state. |

**Bits 21:16 – RXFTHRES[5:0]** Receive FIFO Threshold

| Value | Description |
|---|---|
| 0–32 | Defines the Receive FIFO threshold value (number of data). The FLEX_US_FESR.RXFTHF flag will be set when Receive FIFO goes from “below” threshold state to “equal to or above” threshold state. |

**Bits 13:8 – TXFTHRES[5:0]** Transmit FIFO Threshold

| Value | Description |
|---|---|
| 0–32 | Defines the Transmit FIFO threshold value (number of data). The FLEX_US_FESR.TXFTHF flag will be set when Transmit FIFO goes from “above” threshold state to “equal to or below” threshold state. |

**Bit 7 – FRTSC** FIFO RTS Pin Control enable (Hardware Handshaking mode only)
See Hardware Handshaking for details.

| Value | Description |
|---|---|
| 0 | RTS pin is not controlled by Receive FIFO thresholds. |
| 1 | RTS pin is controlled by Receive FIFO thresholds. |

**Bits 5:4 – RXRDYM[1:0]** Receiver Ready Mode
If FIFOs are enabled, the FLEX_US_CSR.RXRDY flag behaves as follows.

| Value | Name | Description |
|---|---|---|
| 0 | ONE_DATA | RXRDY will be at level '1' when at least one unread data is in the Receive FIFO. |
| 1 | TWO_DATA | RXRDY will be at level '1' when at least two unread data are in the Receive FIFO. |
| 2 | FOUR_DATA | RXRDY will be at level '1' when at least four unread data are in the Receive FIFO. |

**Bits 1:0 – TXRDYM[1:0]** Transmitter Ready Mode
If FIFOs are enabled, the FLEX_US_CSR.TXRDY flag behaves as follows.

| Value | Name | Description |
|---|---|---|
| 0 | ONE_DATA | TXRDY will be at level '1' when at least one data can be written in the Transmit FIFO |
| 1 | TWO_DATA | TXRDY will be at level '1' when at least two data can be written in the Transmit FIFO |
| 2 | FOUR_DATA | TXRDY will be at level '1' when at least four data can be written in the Transmit FIFO |

### 46.10.36 USART FIFO Level Register

**Name:** FLEX_US_FLR
**Offset:** 0x2A4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | RXFL[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TXFL[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 21:16 – RXFL[5:0]** Receive FIFO Level

| Value | Description |
|---|---|
| 0 | There is no unread data in the Receive FIFO. |
| 1-32 | Indicates the number of unread data in the Receive FIFO. |

**Bits 5:0 – TXFL[5:0]** Transmit FIFO Level

| Value | Description |
|---|---|
| 0 | There is no data in the Transmit FIFO. |
| 1-32 | Indicates the number of data in the Transmit FIFO. |

### 46.10.37 USART FIFO Interrupt Enable Register

**Name:** FLEX_US_FIER
**Offset:** 0x2A8
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXFTHF2 | |
| Access | | | | | | | W | |
| Reset | | | | | | | – | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 9 – RXFTHF2** RXFTHF2 Interrupt Enable

**Bit 7 – RXFPTEF** RXFPTEF Interrupt Enable

**Bit 6 – TXFPTEF** TXFPTEF Interrupt Enable

**Bit 5 – RXFTHF** RXFTHF Interrupt Enable

**Bit 4 – RXFFF** RXFFF Interrupt Enable

**Bit 3 – RXFEF** RXFEF Interrupt Enable

**Bit 2 – TXFTHF** TXFTHF Interrupt Enable

**Bit 1 – TXFFF** TXFFF Interrupt Enable

**Bit 0 – TXFEF** TXFEF Interrupt Enable

© 2018 Microchip Technology Inc.

### 46.10.38 USART FIFO Interrupt Disable Register

**Name:** FLEX_US_FIDR
**Offset:** 0x2AC
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXFTHF2 | |
| Access | | | | | | | W | |
| Reset | | | | | | | – | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 9 – RXFTHF2** RXFTHF2 Interrupt Disable

**Bit 7 – RXFPTEF** RXFPTEF Interrupt Disable

**Bit 6 – TXFPTEF** TXFPTEF Interrupt Disable

**Bit 5 – RXFTHF** RXFTHF Interrupt Disable

**Bit 4 – RXFFF** RXFFF Interrupt Disable

**Bit 3 – RXFEF** RXFEF Interrupt Disable

**Bit 2 – TXFTHF** TXFTHF Interrupt Disable

**Bit 1 – TXFFF** TXFFF Interrupt Disable

**Bit 0 – TXFEF** TXFEF Interrupt Disable

### 46.10.39 USART FIFO Interrupt Mask Register

**Name:** FLEX_US_FIMR
**Offset:** 0x2B0
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXFTHF2 | |
| Access | | | | | | | R | |
| Reset | | | | | | | 0 | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 9 – RXFTHF2** RXFTHF2 Interrupt Mask

**Bit 7 – RXFPTEF** RXFPTEF Interrupt Mask

**Bit 6 – TXFPTEF** TXFPTEF Interrupt Mask

**Bit 5 – RXFTHF** RXFTHF Interrupt Mask

**Bit 4 – RXFFF** RXFFF Interrupt Mask

**Bit 3 – RXFEF** RXFEF Interrupt Mask

**Bit 2 – TXFTHF** TXFTHF Interrupt Mask

**Bit 1 – TXFFF** TXFFF Interrupt Mask

**Bit 0 – TXFEF** TXFEF Interrupt Mask

© 2018 Microchip Technology Inc.

### 46.10.40 USART FIFO Event Status Register

**Name:** FLEX_US_FESR
**Offset:** 0x2B4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXFTHF2 | TXFLOCK |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 9 – RXFTHF2** Receive FIFO Threshold Flag 2 (cleared by writing the FLEX_US_CR.RSTSTA bit)

| Value | Description |
|---|---|
| 0 | Number of unread data in Receive FIFO is above RXFTHRES threshold. |
| 1 | Number of unread data in Receive FIFO has reached RXFTHRES2 threshold since the last RSTSTA command was issued. |

**Bit 8 – TXFLOCK** Transmit FIFO Lock

| Value | Description |
|---|---|
| 0 | The Transmit FIFO is not locked. |
| 1 | The Transmit FIFO is locked. |

**Bit 7 – RXFPTEF** Receive FIFO Pointer Error Flag
See 46.7.11.9 FIFO Pointer Error for details.

| Value | Description |
|---|---|
| 0 | No Receive FIFO pointer occurred. |
| 1 | Receive FIFO pointer error occurred. Receiver must be reset. |

**Bit 6 – TXFPTEF** Transmit FIFO Pointer Error Flag
See 46.7.11.9 FIFO Pointer Error for details.

| Value | Description |
|---|---|
| 0 | No Transmit FIFO pointer occurred. |
| 1 | Transmit FIFO pointer error occurred. Transceiver must be reset. |

**Bit 5 – RXFTHF** Receive FIFO Threshold Flag (cleared by writing the FLEX_US_CR.RSTSTA bit)

| Value | Description |
|---|---|
| 0 | Number of unread data in Receive FIFO is below RXFTHRES threshold. |
| 1 | Number of unread data in Receive FIFO has reached RXFTHRES threshold since the last RSTSTA command was issued. |

**Bit 4 – RXFFF** Receive FIFO Full Flag (cleared by writing the FLEX_US_CR.RSTSTA bit)

| Value | Description |
|---|---|
| 0 | Receive FIFO is not empty. |
| 1 | Receive FIFO has been filled since the last RSTSTA command was issued. |

**Bit 3 – RXFEF** Receive FIFO Empty Flag (cleared by writing the FLEX_US_CR.RSTSTA bit)

| Value | Description |
|---|---|
| 0 | Receive FIFO is not empty. |
| 1 | Receive FIFO has been emptied since the last RSTSTA command was issued. |

**Bit 2 – TXFTHF** Transmit FIFO Threshold Flag (cleared by writing the FLEX_US_CR.RSTSTA bit)

| Value | Description |
|---|---|
| 0 | Number of data in Transmit FIFO is above TXFTHRES threshold. |
| 1 | Number of data in Transmit FIFO has reached TXFTHRES threshold since the last RSTSTA command was issued. |

**Bit 1 – TXFFF** Transmit FIFO Full Flag (cleared by writing the FLEX_US_CR.RSTSTA bit)

| Value | Description |
|---|---|
| 0 | Transmit FIFO is not full. |
| 1 | Transmit FIFO has been filled since the last RSTSTA command was issued. |

**Bit 0 – TXFEF** Transmit FIFO Empty Flag (cleared by writing the FLEX_US_CR.RSTSTA bit)

| Value | Description |
|---|---|
| 0 | Transmit FIFO is not empty. |
| 1 | Transmit FIFO has been emptied since the last RSTSTA command was issued. |

### 46.10.41 USART Write Protection Mode Register

**Name:** FLEX_US_WPMR
**Offset:** 0x2E4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x555341 | PASSWD | Writing any other value in this field aborts the write operation of bit WPEN. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See USART Register Write Protection for the list of registers that can be write-protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection on configuration registers if WPKEY corresponds to 0x555341 (“USA” in ASCII). |
| 1 | Enables the write protection on configuration registers if WPKEY corresponds to 0x555341 (“USA” in ASCII). |

### 46.10.42 USART Write Protection Status Register

**Name:** FLEX_US_WPSR
**Offset:** 0x2E8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of FLEX_US_WPSR. |
| 1 | A write protection violation has occurred since the last read of FLEX_US_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

### 46.10.43 SPI Control Register

**Name:** FLEX_SPI_CR
**Offset:** 0x400
**Reset:** –
**Property:** Write-only

This register can only be written if the WPCREN bit is cleared in the SPI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FIFODIS | FIFOEN | | | | | | LASTXFER |
| Access | W | W | | | | | | W |
| Reset | – | – | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXFCLR | TXFCLR |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | REQCLR | | | | |
| Access | | | | W | | | | |
| Reset | | | | – | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWRST | | | | | | SPIDIS | SPIEN |
| Access | W | | | | | | W | W |
| Reset | – | | | | | | – | – |

**Bit 31 – FIFODIS** FIFO Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Transmit and Receive FIFOs |

**Bit 30 – FIFOEN** FIFO Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Transmit and Receive FIFOs |

**Bit 24 – LASTXFER** Last Transfer
See Peripheral Selection for more details.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The current NPCS will be de-asserted after the character written in TD has been transferred. When CSAAT is set, the communication with the current serial peripheral can be closed by raising the corresponding NPCS line as soon as TD transfer is completed. |

**Bit 17 – RXFCLR** Receive FIFO Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Empties the Receive FIFO. |

**Bit 16 – TXFCLR** Transmit FIFO Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Empties the Transmit FIFO. |

**Bit 12 – REQCLR** Request to Clear the Comparison Trigger
SleepWalking enabled:

0: No effect.

1: Clears the potential clock request currently issued by SPI, thus the potential system wakeup is cancelled.

SleepWalking disabled:

0: No effect.

1: Restarts the comparison trigger to enable FLEX_SPI_RDR loading.

**Bit 7 – SWRST** SPI Software Reset
The SPI is in Slave mode after software reset.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the SPI. A software-triggered hardware reset of the SPI interface is performed. |

**Bit 1 – SPIDIS** SPI Disable
If a transfer is in progress when SPIDIS is set, the SPI completes the transmission of the shifter register and does not start any new transfer, even if the FLEX_US_THR is loaded.

All pins are set in Input mode after completion of the transmission in progress, if any.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the SPI. |

**Bit 0 – SPIEN** SPI Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the SPI to transfer and receive data. |

### 46.10.44 SPI Mode Register

**Name:** FLEX_SPI_MR
**Offset:** 0x404
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the SPI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DLYBCS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | PCS[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | CMPMODE | | | | |
| Access | | | | R/W | | | | |
| Reset | | | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LLB | | WDRBT | MODFDIS | BRSRCCLK | PCSDEC | PS | MSTR |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – DLYBCS[7:0]** Delay Between Chip Selects
This field defines the delay between the inactivation and the activation of NPCS. The DLYBCS time guarantees nonoverlapping chip selects and solves bus contentions in case of peripherals having long data float times.

If DLYBCS is ≤ 6, six peripheral clock periods are inserted by default.

Otherwise, the following equations determine the delay:

If FLEX_SPI_MR.BRSRCCLK = 0: DLYBCS = Delay Between Chip Selects × $f_{peripheral\ clock}$

If FLEX_SPI_MR.BRSRCCLK = 1: DLYBCS = Delay Between Chip Selects × $f_{GCLK}$

**Bits 17:16 – PCS[1:0]** Peripheral Chip Select
This field is only used if fixed peripheral select is active (PS = 0).

If PCSDEC = 0:

PCS = x0 NPCS[1:0] = 10

PCS = 01 NPCS[1:0] = 01

PCS = 11 forbidden (no peripheral is selected)

(x = don’t care)

If PCSDEC = 1:

NPCS[1:0] output signals = PCS

**Bit 12 – CMPMODE** Comparison Mode

| Value | Name | Description |
|---|---|---|
| 0 | FLAG_ONLY | Any character is received and comparison function drives CMP flag. |
| 1 | START_CONDITION | Comparison condition must be met to start reception of all incoming characters until REQCLR is set. |

**Bit 7 – LLB** Local Loopback Enable
LLB controls the local loopback on the data shift register for testing in Master mode only (MISO is internally connected on MOSI).

| Value | Description |
|---|---|
| 0 | Local loopback path disabled. |
| 1 | Local loopback path enabled. |

**Bit 5 – WDRBT** Wait Data Read Before Transfer

| Value | Description |
|---|---|
| 0 | No Effect. In Master mode, a transfer can be initiated regardless of the FLEX_SPI_RDR state. |
| 1 | In Master mode, a transfer can start only if FLEX_SPI_RDR is empty, i.e., does not contain any unread data. This mode prevents overrun error in reception. |

**Bit 4 – MODFDIS** Mode Fault Detection

| Value | Description |
|---|---|
| 0 | Mode fault detection is enabled. |
| 1 | Mode fault detection is disabled. |

**Bit 3 – BRSRCCLK** Bit Rate Source Clock
If the bit BRSRCCLK = 1, the FLEX_US_CSRx.SCBR field must be programmed with a value greater than 1.

| Value | Name | Description |
|---|---|---|
| 0 | PERIPH_CLK | The peripheral clock is the source clock for the bit rate generation. |
| 1 | GCLK | GCLK is the source clock for the bit rate generation, thus the bit rate can be independent of the core/peripheral clock. |

**Bit 2 – PCSDEC** Chip Select Decode
When PCSDEC equals one, up to 3 Chip Select signals can be generated with the two NPCS lines using an external 2- to 4-bit decoder. The Chip Select registers define the characteristics of the 3 chip selects, with the following rules:

FLEX_SPI_CSR0 defines peripheral chip select signals 0 to 1.

FLEX_SPI_CSR1 defines peripheral chip select signal 2.

| Value | Description |
|---|---|
| 0 | The chip selects are directly connected to a peripheral device. |

| Value | Description |
|---|---|
| 1 | The two NPCS chip select lines are connected to a 2- to 4-bit decoder. |

**Bit 1 – PS** Peripheral Select

| Value | Description |
|---|---|
| 0 | Fixed Peripheral Select |
| 1 | Variable Peripheral Select |

**Bit 0 – MSTR** Master/Slave Mode

| Value | Description |
|---|---|
| 0 | SPI is in Slave mode. |
| 1 | SPI is in Master mode. |

### 46.10.45 SPI Receive Data Register

**Name:** FLEX_SPI_RDR
**Offset:** 0x408
**Reset:** 0x00000000
**Property:** Read-only

If FIFO is enabled (FLEX_SPI_CR.FIFOEN) and FLEX_SPI_FMR.RXRDYM = 0, see SPI Single Data Mode for details.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PCS[3:0] | | | |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 19:16 – PCS[3:0]** Peripheral Chip Select
In Master mode only, these bits indicate the value on the NPCS pins at the end of a transfer. Otherwise, these bits are read as zero.

**Note:** When using Variable Peripheral Select mode (FLEX_SPI_MR.PS = 1), it is mandatory to set the FLEX_SPI_MR.WDRBT bit to 1 if the PCS field must be processed in FLEX_SPI_RDR.

**Bits 15:0 – RD[15:0]** Receive Data
Data received by the SPI Interface is stored in this register in a right-justified format. Unused bits are read as zero.

### 46.10.46 SPI Receive Data Register (FIFO Multiple Data, 8-bit)

**Name:** FLEX_SPI_RDR (FIFO_MULTI_DATA_8)
**Offset:** 0x408
**Reset:** 0x00000000
**Property:** Read-only

If FIFO is enabled (FLEX_SPI_CR.FIFOEN) and FLEX_SPI_FMR.RXRDYM > 0, see SPI Multiple Data Mode for details.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RD3[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RD2[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RD1[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RD0[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0:7, 8:15, 16:23, 24:31 – RDx** Receive Data
First unread data in the Receive FIFO. Data received by the SPI Interface is stored in this register in a right-justified format. Unused bits are read as zero.

### 46.10.47 SPI Receive Data Register (FIFO Multiple Data, 16-bit)

**Name:** FLEX_SPI_RDR (FIFO_MULTI_DATA_16)
**Offset:** 0x408
**Reset:** 0x00000000
**Property:** Read-only

If FIFO is enabled (FLEX_SPI_CR.FIFOEN) and FLEX_SPI_FMR.RXRDYM > 0, see SPI Multiple Data Mode for details.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RD1[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RD1[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RD0[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RD0[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0:15, 16:31 – RDx** Receive Data
First unread data in the Receive FIFO. Data received by the SPI Interface is stored in this register in a right-justified format. Unused bits are read as zero.

### 46.10.48 SPI Transmit Data Register

**Name:** FLEX_SPI_TDR
**Offset:** 0x40C
**Reset:** –
**Property:** Write-only

If FIFO is enabled (FLEX_SPI_CR.FIFOEN) and FLEX_SPI_FMR.TXRDYM = 0, see 46.8.7.6 SPI Single Data Mode for details.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | LASTXFER |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PCS[3:0] | | | |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TD[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TD[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 24 – LASTXFER** Last Transfer
This field is only used if variable peripheral select is active (FLEX_SPI_MR.PS = 1).

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The current NPCS is de-asserted after the transfer of the character written in TD. When FLEX_SPI_CSRx.CSAAT is set, the communication with the current serial peripheral can be closed by raising the corresponding NPCS line as soon as TD transfer is completed. |

**Bits 19:16 – PCS[3:0]** Peripheral Chip Select
This field is only used if variable peripheral select is active (FLEX_SPI_MR.PS = 1).

If FLEX_SPI_MR.PCSDEC = 0:

PCS = x0 NPCS[1:0] = 10

PCS = 01 NPCS[1:0] = 01

PCS = 11 forbidden (no peripheral is selected)

(x = don’t care)

If FLEX_SPI_MR.PCSDEC = 1:

NPCS[1:0] output signals = PCS

**Bits 15:0 – TD[15:0]** Transmit Data
Data to be transmitted by the SPI Interface is stored in this register. Information to be transmitted must be written to the transmit data register in a right-justified format.

### 46.10.49 SPI Transmit Data Register (FIFO Multiple Data, 8- to 16-bit)

**Name:** FLEX_SPI_TDR (FIFO_MULTI_DATA)
**Offset:** 0x40C
**Reset:** –
**Property:** Write-only

If FIFO is enabled (FLEX_SPI_CR.FIFOEN) and FLEX_SPI_FMR.TXRDYM > 0, see Section 1.8.7.7 “SPI Multiple Data Mode” for details.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TD1[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TD1[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TD0[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TD0[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0:15, 16:31 – TDx** Transmit Data
Next data to write in the Transmit FIFO. Information to be transmitted must be written to this register in a right-justified format.

### 46.10.50 SPI Status Register

**Name:** FLEX_SPI_SR
**Offset:** 0x410
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | | | | | | SPIENS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | | | | | CMP | UNDES | TXEMPTY | NSSR |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | OVRES | MODF | TDRE | RDRF |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 31 – RXFPTEF** Receive FIFO Pointer Error Flag
See 46.8.7.8 FIFO Pointer Error for details.

| Value | Description |
|---|---|
| 0 | No Receive FIFO pointer occurred |
| 1 | Receive FIFO pointer error occurred. Receiver must be reset |

**Bit 30 – TXFPTEF** Transmit FIFO Pointer Error Flag
See 46.8.7.8 FIFO Pointer Error for details.

| Value | Description |
|---|---|
| 0 | No Transmit FIFO pointer occurred |
| 1 | Transmit FIFO pointer error occurred. Transceiver must be reset |

**Bit 29 – RXFTHF** Receive FIFO Threshold Flag

| Value | Description |
|---|---|
| 0 | Number of unread data in Receive FIFO is below RXFTHRES threshold or RXFTH flag has been cleared. |
| 1 | Number of unread data in Receive FIFO has reached RXFTHRES threshold (changing states from “below threshold” to “equal to or above threshold”). |

**Bit 28 – RXFFF** Receive FIFO Full Flag

| Value | Description |
|---|---|
| 0 | Receive FIFO is not empty or RXFE flag has been cleared. |
| 1 | Receive FIFO has been filled (changing states from “not full” to “full”). |

**Bit 27 – RXFEF** Receive FIFO Empty Flag

| Value | Description |
|---|---|
| 0 | Receive FIFO is not empty or RXFE flag has been cleared. |
| 1 | Receive FIFO has been emptied (changing states from “not empty” to “empty”). |

**Bit 26 – TXFTHF** Transmit FIFO Threshold Flag (cleared on read)

| Value | Description |
|---|---|
| 0 | Number of data in Transmit FIFO is above TXFTHRES threshold. |
| 1 | Number of data in Transmit FIFO has reached TXFTHRES threshold since the last read of FLEX_SPI_SR. |

**Bit 25 – TXFFF** Transmit FIFO Full Flag (cleared on read)

| Value | Description |
|---|---|
| 0 | Transmit FIFO is not full or TXFF flag has been cleared. |
| 1 | Transmit FIFO has been filled since the last read of FLEX_SPI_SR. |

**Bit 24 – TXFEF** Transmit FIFO Empty Flag (cleared on read)

| Value | Description |
|---|---|
| 0 | Transmit FIFO is not empty. |
| 1 | Transmit FIFO has been emptied since the last read of FLEX_SPI_SR. |

**Bit 16 – SPIENS** SPI Enable Status

| Value | Description |
|---|---|
| 0 | SPI is disabled. |
| 1 | SPI is enabled. |

**Bit 11 – CMP** Comparison Status (cleared on read)

| Value | Description |
|---|---|
| 0 | No received character matched the comparison criteria programmed in VAL1 and VAL2 fields in FLEX_SPI_CMPR since the last read of FLEX_SPI_SR. |
| 1 | A received character matched the comparison criteria since the last read of FLEX_SPI_SR. |

**Bit 10 – UNDES** Underrun Error Status (Slave mode only) (cleared on read)

| Value | Description |
|---|---|
| 0 | No underrun has been detected since the last read of FLEX_SPI_SR. |
| 1 | A transfer starts whereas no data has been loaded in FLEX_SPI_TDR, cleared when FLEX_SPI_SR is read. |

**Bit 9 – TXEMPTY** Transmission Registers Empty (cleared by writing FLEX_SPI_TDR)

| Value | Description |
|---|---|
| 0 | As soon as data is written in FLEX_SPI_TDR. |
| 1 | FLEX_SPI_TDR and internal shift register are empty. If a transfer delay has been defined, TXEMPTY is set after the end of this delay. |

**Bit 8 – NSSR** NSS Rising (cleared on read)

| Value | Description |
|---|---|
| 0 | No rising edge detected on NSS pin since the last read of FLEX_SPI_SR. |
| 1 | A rising edge occurred on NSS pin since the last read of FLEX_SPI_SR. |

**Bit 3 – OVRES** Overrun Error Status (cleared on read)
An overrun occurs when FLEX_SPI_RDR is loaded at least twice from the shift register since the last read of FLEX_SPI_RDR.

| Value | Description |
|---|---|
| 0 | No overrun has been detected since the last read of FLEX_SPI_SR. |
| 1 | An overrun has occurred since the last read of FLEX_SPI_SR. |

**Bit 2 – MODF** Mode Fault Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No mode fault has been detected since the last read of FLEX_SPI_SR. |
| 1 | A mode fault occurred since the last read of FLEX_SPI_SR. |

**Bit 1 – TDRE** Transmit Data Register Empty (cleared by writing FLEX_SPI_TDR)
When FIFOs are disabled:

0: Data has been written to FLEX_SPI_TDR and not yet transferred to the internal shift register.

1: The last data written to FLEX_SPI_TDR has been transferred to the internal shift register.

TDRE is cleared when the SPI is disabled or at reset. Enabling the SPI sets the TDRE flag.

When FIFOs are enabled:

0: Transmit FIFO cannot accept more data.

1: Transmit FIFO can accept data; one or more data can be written according to TXRDYM field configuration.

TDRE behavior with FIFOs enabled is illustrated in 46.8.7.5 TXEMPTY, TDRE and RDRF Behavior.

**Bit 0 – RDRF** Receive Data Register Full (cleared by reading FLEX_SPI_RDR)
When FIFOs are disabled:

0: No data has been received since the last read of FLEX_SPI_RDR.

1: Data has been received and the received data has been transferred from the internal shift register to FLEX_SPI_RDR since the last read of FLEX_SPI_RDR.

When FIFOs are enabled:

0: Receive FIFO is empty; no data to read.

1: At least one unread data is in the Receive FIFO.

RDRF behavior with FIFOs enabled is illustrated in 46.8.7.5 TXEMPTY, TDRE and RDRF Behavior.

### 46.10.51 SPI Interrupt Enable Register

**Name:** FLEX_SPI_IER
**Offset:** 0x414
**Reset:** –
**Property:** Write-only

This register can only be written if the WPITEN bit is cleared in the SPI Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CMP | UNDES | TXEMPTY | NSSR |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | MODF | TDRE | RDRF |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

**Bit 31 – RXFPTEF** RXFPTEF Interrupt Enable

**Bit 30 – TXFPTEF** TXFPTEF Interrupt Enable

**Bit 29 – RXFTHF** RXFTHF Interrupt Enable

**Bit 28 – RXFFF** RXFFF Interrupt Enable

**Bit 27 – RXFEF** RXFEF Interrupt Enable

**Bit 26 – TXFTHF** TXFTHF Interrupt Enable

**Bit 25 – TXFFF** TXFFF Interrupt Enable

**Bit 24 – TXFEF** TXFEF Interrupt Enable

**Bit 11 – CMP** Comparison Interrupt Enable

**Bit 10 – UNDES** Underrun Error Interrupt Enable

**Bit 9 – TXEMPTY** Transmission Registers Empty Enable

**Bit 8 – NSSR** NSS Rising Interrupt Enable

**Bit 3 – OVRES** Overrun Error Interrupt Enable

**Bit 2 – MODF** Mode Fault Error Interrupt Enable

**Bit 1 – TDRE** SPI Transmit Data Register Empty Interrupt Enable

**Bit 0 – RDRF** Receive Data Register Full Interrupt Enable

© 2018 Microchip Technology Inc.

### 46.10.52 SPI Interrupt Disable Register

**Name:** FLEX_SPI_IDR
**Offset:** 0x418
**Reset:** –
**Property:** Write-only

This register can only be written if the WPITEN bit is cleared in the SPI Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CMP | UNDES | TXEMPTY | NSSR |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | MODF | TDRE | RDRF |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

**Bit 31 – RXFPTEF** RXFPTEF Interrupt Disable

**Bit 30 – TXFPTEF** TXFPTEF Interrupt Disable

**Bit 29 – RXFTHF** RXFTHF Interrupt Disable

**Bit 28 – RXFFF** RXFFF Interrupt Disable

**Bit 27 – RXFEF** RXFEF Interrupt Disable

**Bit 26 – TXFTHF** TXFTHF Interrupt Disable

**Bit 25 – TXFFF** TXFFF Interrupt Disable

**Bit 24 – TXFEF** TXFEF Interrupt Disable

**Bit 11 – CMP** Comparison Interrupt Disable

**Bit 10 – UNDES** Underrun Error Interrupt Disable

**Bit 9 – TXEMPTY** Transmission Registers Empty Disable

**Bit 8 – NSSR** NSS Rising Interrupt Disable

**Bit 3 – OVRES** Overrun Error Interrupt Disable

**Bit 2 – MODF** Mode Fault Error Interrupt Disable

**Bit 1 – TDRE** SPI Transmit Data Register Empty Interrupt Disable

**Bit 0 – RDRF** Receive Data Register Full Interrupt Disable

### 46.10.53 SPI Interrupt Mask Register

**Name:** FLEX_SPI_IMR
**Offset:** 0x41C
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CMP | UNDES | TXEMPTY | NSSR |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | MODF | TDRE | RDRF |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 31 – RXFPTEF** RXFPTEF Interrupt Mask

**Bit 30 – TXFPTEF** TXFPTEF Interrupt Mask

**Bit 29 – RXFTHF** RXFTHF Interrupt Mask

**Bit 28 – RXFFF** RXFFF Interrupt Mask

**Bit 27 – RXFEF** RXFEF Interrupt Mask

**Bit 26 – TXFTHF** TXFTHF Interrupt Mask

**Bit 25 – TXFFF** TXFFF Interrupt Mask

**Bit 24 – TXFEF** TXFEF Interrupt Mask

**Bit 11 – CMP** Comparison Interrupt Mask

**Bit 10 – UNDES** Underrun Error Interrupt Mask

**Bit 9 – TXEMPTY** Transmission Registers Empty Mask

**Bit 8 – NSSR** NSS Rising Interrupt Mask

**Bit 3 – OVRES** Overrun Error Interrupt Mask

**Bit 2 – MODF** Mode Fault Error Interrupt Mask

**Bit 1 – TDRE** SPI Transmit Data Register Empty Interrupt Mask

**Bit 0 – RDRF** Receive Data Register Full Interrupt Mask

### 46.10.54 SPI Chip Select Register

**Name:** FLEX_SPI_CSRx
**Offset:** 0x0430 + x*0x04 [x=0..1]
**Reset:** 0x00000000
**Property:** R/W

This register can only be written if the WPEN bit is cleared in the SPI Write Protection Mode Register.

FLEX_SPI_CSRx must be written even if the user wants to use the default reset values. The BITS field is not updated with the translated value unless the register is written.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DLYBCT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DLYBS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SCBR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BITS[3:0] | | | | CSAAT | CSNAAT | NCPHA | CPOL |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – DLYBCT[7:0]** Delay Between Consecutive Transfers
This field defines the delay between two consecutive transfers with the same peripheral without removing the chip select. The delay is always inserted after each transfer and before removing the chip select if needed.

When DLYBCT = 0, no delay between consecutive transfers is inserted and the clock keeps its duty cycle over the character transfers.

Otherwise, the following equations determine the delay:

If FLEX_SPI_MR.BRSRCCLK = 0: DLYBCT = Delay Between Consecutive Transfers × $f_{peripheral\ clock}$ / 32

If FLEX_SPI_MR.BRSRCCLK = 1: DLYBCT = Delay Between Consecutive Transfers × $f_{GCLK}$ / 32

**Bits 23:16 – DLYBS[7:0]** Delay Before SPCK
This field defines the delay from NPCS falling edge (activation) to the first valid SPCK transition.

When DLYBS = 0, the delay is half the SPCK clock period.

Otherwise, the following equations determine the delay:

If FLEX_SPI_MR.BRSRCCLK = 0: DLYBS = Delay Before SPCK × $f_{peripheral\ clock}$

If FLEX_SPI_MR.BRSRCCLK = 1: DLYBS = Delay Before SPCK × $f_{GCLK}$

**Bits 15:8 – SCBR[7:0]** Serial Clock Bit Rate
In Master mode, the SPI Interface uses a modulus counter to derive the SPCK bit rate from the clock defined by the bit BRSRCCLK. The bit rate is selected by writing a value from 1 to 255 in the SCBR field. The following equations determine the SPCK bit rate:

If FLEX_SPI_MR.BRSRCCLK = 0: SCBR = $f_{peripheral\ clock}$ / SPCK Bit Rate

If FLEX_SPI_MR.BRSRCCLK = 1: SCBR = $f_{GCLK}$ / SPCK Bit Rate

Programming the SCBR field to 0 is forbidden. Triggering a transfer while SCBR is at 0 can lead to unpredictable results.

If BRSRCCLK = 1 in FLEX_SPI_MR, SCBR must be programmed with a value greater than 1.

At reset, SCBR is 0 and the user has to program it at a valid value before performing the first transfer.

**Note:** If one of the FLEX_SPI_CSRx.SCBR fields is set to 1, the other FLEX_SPI_CSRx.SCBR fields must be set to 1 as well, if they are used to process transfers. If they are not used to transfer data, they can be set at any value.

**Bits 7:4 – BITS[3:0]** Bits Per Transfer
See Note.

The BITS field determines the number of data bits transferred. Reserved values should not be used.

| Value | Name | Description |
|---|---|---|
| 0 | 8_BIT | 8 bits for transfer |
| 1 | 9_BIT | 9 bits for transfer |
| 2 | 10_BIT | 10 bits for transfer |
| 3 | 11_BIT | 11 bits for transfer |
| 4 | 12_BIT | 12 bits for transfer |
| 5 | 13_BIT | 13 bits for transfer |
| 6 | 14_BIT | 14 bits for transfer |
| 7 | 15_BIT | 15 bits for transfer |
| 8 | 16_BIT | 16 bits for transfer |
| 9–15 | Reserved | |

**Bit 3 – CSAAT** Chip Select Active After Transfer

| Value | Description |
|---|---|
| 0 | The Peripheral Chip Select Line rises as soon as the last transfer is achieved. |
| 1 | The Peripheral Chip Select does not rise after the last transfer is achieved. It remains active until a new transfer is requested on a different chip select. |

**Bit 2 – CSNAAT** Chip Select Not Active After Transfer (Ignored if CSAAT = 1)

If FLEX_SPI_MR.BRSRCCLK = 0: $\frac{DLYBCS}{f_{peripheral\ clock}}$ (if DLYBCS ≠ 0)

If FLEX_SPI_MR.BRSRCCLK = 1: $\frac{DLYBCS}{f_{GCLK}}$

If DLYBCS < 6, a minimum of six periods is introduced.

| Value | Description |
|---|---|
| 0 | The Peripheral Chip Select does not rise between two transfers if the FLEX_SPI_TDR is reloaded before the end of the first transfer and if the two transfers occur on the same Chip Select. |
| 1 | The Peripheral Chip Select rises systematically after each transfer performed on the same slave. It remains inactive after the end of transfer for a minimal duration of: |

**Bit 1 – NCPHA** Clock Phase
NCPHA determines which edge of SPCK causes data to change and which edge causes data to be captured. NCPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.

| Value | Description |
|---|---|
| 0 | Data are changed on the leading edge of SPCK and captured on the following edge of SPCK. |
| 1 | Data are captured on the leading edge of SPCK and changed on the following edge of SPCK. |

**Bit 0 – CPOL** Clock Polarity
CPOL is used to determine the inactive state value of the serial clock (SPCK). It is used with NCPHA to produce the required clock/data relationship between master and slave devices.

| Value | Description |
|---|---|
| 0 | The inactive state value of SPCK is logic level zero. |
| 1 | The inactive state value of SPCK is logic level one. |

### 46.10.55 SPI FIFO Mode Register

**Name:** FLEX_SPI_FMR
**Offset:** 0x440
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | RXFTHRES[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | TXFTHRES[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | RXRDYM[1:0] | | | | TXRDYM[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bits 29:24 – RXFTHRES[5:0]** Receive FIFO Threshold

| Value | Description |
|---|---|
| 0-32 | Defines the Receive FIFO threshold value (number of data). The FLEX_SPI_SR.RXFTH flag will be set when Receive FIFO goes from “below” threshold state to “equal to or above” threshold state. |

**Bits 21:16 – TXFTHRES[5:0]** Transmit FIFO Threshold

| Value | Description |
|---|---|
| 0-32 | Defines the Transmit FIFO threshold value (number of data). The FLEX_SPI_SR.TXFTH flag will be set when Transmit FIFO goes from “above” threshold state to “equal to or below” threshold state. |

**Bits 5:4 – RXRDYM[1:0]** Receive Data Register Full Mode
If FIFOs are enabled, the FLEX_SPI_SR.RDRF flag behaves as follows.

| Value | Name | Description |
|---|---|---|
| 0 | ONE_DATA | RDRF will be at level ‘1’ when at least one unread data is in the Receive FIFO. |

| Value | Name | Description |
|---|---|---|
| 1 | TWO_DATA | RDRF will be at level ‘1’ when at least two unread data are in the Receive FIFO.<br>Cannot be used when FLEX_SPI_MR.MSTR =1, or if FLEX_SPI_MR.PS =1. |
| 2 | FOUR_DATA | RDRF will be at level ‘1’ when at least four unread data are in the Receive FIFO.<br>Cannot be used when FLEX_SPI_CSRx.BITS is greater than 0, or if FLEX_SPI_MR.MSTR =1, or if FLEX_SPI_MR.PS =1. |

**Bits 1:0 – TXRDYM[1:0]** Transmit Data Register Empty Mode
If FIFOs are enabled, the FLEX_SPI_SR.TDRE flag behaves as follows.

| Value | Name | Description |
|---|---|---|
| 0 | ONE_DATA | TDRE will be at level ‘1’ when at least one data can be written in the Transmit FIFO. |
| 1 | TWO_DATA | TDRE will be at level ‘1’ when at least two data can be written in the Transmit FIFO.<br>Cannot be used if FLEX_SPI_MR.PS =1. |

### 46.10.56 SPI FIFO Level Register

**Name:** FLEX_SPI_FLR
**Offset:** 0x444
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | RXFL[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TXFL[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 21:16 – RXFL[5:0]** Receive FIFO Level

| Value | Description |
|---|---|
| 0 | There is no unread data in the Receive FIFO. |
| 1-32 | Indicates the number of unread data in the Receive FIFO. |

**Bits 5:0 – TXFL[5:0]** Transmit FIFO Level

| Value | Description |
|---|---|
| 0 | There is no data in the Transmit FIFO. |
| 1-32 | Indicates the number of data in the Transmit FIFO. |

### 46.10.57 SPI Comparison Register

**Name:** FLEX_SPI_CMPR
**Offset:** 0x448
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the SPI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | VAL2[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | VAL2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | VAL1[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | VAL1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – VAL2[15:0]** Second Comparison Value for Received Character

| Value | Description |
|---|---|
| 0-65535 | The received character must be lower than or equal to the value of VAL2 and higher than or equal to VAL1 to set the FLEX_SPI_CSR.CMP flag. If asynchronous partial wakeup (SleepWalking) is enabled in PMC_SLPWK_ER, the SPI requests a system wakeup if condition is met. |

**Bits 15:0 – VAL1[15:0]** First Comparison Value for Received Character

| Value | Description |
|---|---|
| 0-65535 | The received character must be higher than or equal to the value of VAL1 and lower than or equal to VAL2 to set the FLEX_SPI_SR.CMP flag. If asynchronous partial wakeup (SleepWalking) is enabled in PMC_SLPWK_ER, the SPI requests a system wakeup if the condition is met. |

### 46.10.58 SPI Write Protection Mode Register

**Name:** FLEX_SPI_WPMR
**Offset:** 0x4E4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | WPCREN | WPITEN | WPEN |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x535049 | PASSWD | Writing any other value in this field aborts the write operation of bits WPEN, WPITEN and WPCREN.<br>Always reads as 0. |

**Bit 2 – WPCREN** Write Protection Control Enable

| Value | Description |
|---|---|
| 0 | Disables the write protection on control register if WPKEY corresponds to 0x535049 (“SPI” in ASCII). |
| 1 | Enables the write protection on control register if WPKEY corresponds to 0x535049 (“SPI” in ASCII). |

**Bit 1 – WPITEN** Write Protection Interrupt Enable

| Value | Description |
|---|---|
| 0 | Disables the write protection on interrupt registers if WPKEY corresponds to 0x535049 (“SPI” in ASCII). |
| 1 | Enables the write protection on interrupt registers if WPKEY corresponds to 0x535049 (“SPI” in ASCII). |

**Bit 0 – WPEN** Write Protection Enable
See 46.8.8 SPI Register Write Protection for the list of registers that can be write-protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x535049 (“SPI” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x535049 (“SPI” in ASCII). |

### 46.10.59 SPI Write Protection Status Register

**Name:** FLEX_SPI_WPSR
**Offset:** 0x4E8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 15:8 – WPVSRC[7:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protect violation has occurred since the last read of FLEX_SPI_WPSR. |
| 1 | A write protect violation has occurred since the last read of FLEX_SPI_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

### 46.10.60 TWI Control Register

**Name:** FLEX_TWI_CR
**Offset:** 0x600
**Reset:** –
**Property:** Write-only

This register can only be written if the WPCREN bit is cleared in the TWI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | FIFODIS | FIFOEN | | LOCKCLR | | THRCLR |
| Access | | | W | W | | W | | W |
| Reset | | | – | – | | – | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ACMDIS | ACMEN |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CLEAR | PECRQ | PECDIS | PECEN | SMBDIS | SMBEN | HSDIS | HSEN |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWRST | QUICK | SVDIS | SVEN | MSDIS | MSEN | STOP | START |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 29 – FIFODIS** FIFO Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable the Transmit and Receive FIFOs |

**Bit 28 – FIFOEN** FIFO Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable the Transmit and Receive FIFOs |

**Bit 26 – LOCKCLR** Lock Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clear the TWI FSM lock. |

**Bit 24 – THRCLR** Transmit Holding Register Clear

| Value | Description |
|---|---|
| 0 | No effect. |

| Value | Description |
|---|---|
| 1 | Clear the Transmit Holding Register and set TXRDY, TXCOMP flags. |

**Bit 17 – ACMDIS** Alternative Command Mode Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Alternative Command mode disabled. |

**Bit 16 – ACMEN** Alternative Command Mode Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Alternative Command mode enabled. |

**Bit 15 – CLEAR** Bus CLEAR Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If Master mode is enabled, send a bus clear command. |

**Bit 14 – PECRQ** PEC Request

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | A PEC check or transmission is requested. |

**Bit 13 – PECDIS** Packet Error Checking Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | SMBus PEC (CRC) generation and check disabled. |

**Bit 12 – PECEN** Packet Error Checking Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | SMBus PEC (CRC) generation and check enabled. |

**Bit 11 – SMBDIS** SMBus Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | SMBus mode disabled. |

**Bit 10 – SMBEN** SMBus Mode Enabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If SMBDIS = 0, SMBus mode enabled. |

**Bit 9 – HSDIS** TWI High-Speed Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | High-speed mode disabled. |

**Bit 8 – HSEN** TWI High-Speed Mode Enabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | High-speed mode enabled. |

**Bit 7 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Equivalent to a system reset. |

**Bit 6 – QUICK** SMBus Quick Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If Master mode is enabled, an SMBus Quick Command is sent. |

**Bit 5 – SVDIS** TWI Slave Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Slave mode is disabled. The shifter and holding characters (if it contains data) are transmitted in the case of a read operation. In a write operation, the character being transferred must be completely received before disabling. |

**Bit 4 – SVEN** TWI Slave Mode Enabled
Switching from Master to Slave mode is only permitted when TXCOMP = 1.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Slave mode (SVDIS must be written to 0). |

**Bit 3 – MSDIS** TWI Master Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The Master mode is disabled, all pending data is transmitted. The shifter and holding characters (if it contains data) are transmitted in case of write operation. In read operation, the character being transferred must be completely received before disabling. |

**Bit 2 – MSEN** TWI Master Mode Enabled
Switching from Slave to Master mode is only permitted when TXCOMP = 1.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Master mode (MSDIS must be written to 0). |

**Bit 1 – STOP** Send a STOP Condition

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | STOP condition is sent just after completing the current byte transmission in Master Read mode.<br>– In single data byte master read, both START and STOP must be set.<br>– In multiple data bytes master read, the STOP must be set after the last data received but one.<br>– In Master Read mode, if a NACK bit is received, the STOP is automatically performed.<br>– In master data write operation, a STOP condition will be sent after the transmission of the current data is finished. |

**Bit 0 – START** Send a START Condition
This action is necessary when the TWI peripheral needs to read data from a slave. When configured in Master mode with a write operation, a frame is sent as soon as the user writes a character in the Transmit Holding Register (FLEX_TWI_THR).

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | A frame beginning with a START bit is transmitted according to the features defined in the TWI Master Mode Register (FLEX_TWI_MMR). |

### 46.10.61 TWI Control Register (FIFO_ENABLED)

**Name:** FLEX_TWI_CR (FIFO_ENABLED)
**Offset:** 0x600
**Reset:** –
**Property:** Write-only

If FIFO is enabled (FLEX_US_CR.FIFOEN bit), see 46.9.6.8 TWI Multiple Data Mode for details.

This register can only be written if the WPCREN bit is cleared in the TWI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | FIFODIS | FIFOEN | | TXFLCLR | RXFCLR | TXFCLR |
| Access | | | W | W | | W | W | W |
| Reset | | | – | – | | – | – | – |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | | | | | ACMDIS | ACMEN |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | CLEAR | PECRQ | PECDIS | PECEN | SMBDIS | SMBEN | HSDIS | HSEN |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | SWRST | QUICK | SVDIS | SVEN | MSDIS | MSEN | STOP | START |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 29 – FIFODIS** FIFO Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable the Transmit and Receive FIFOs. |

**Bit 28 – FIFOEN** FIFO Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable the Transmit and Receive FIFOs. |

**Bit 26 – TXFLCLR** Transmit FIFO Lock CLEAR

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears the Transmit FIFO Lock. |

**Bit 25 – RXFCLR** Receive FIFO Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Empties the Receive FIFO. |

**Bit 24 – TXFCLR** Transmit FIFO Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Empties the Transmit FIFO. |

**Bit 17 – ACMDIS** Alternative Command Mode Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Alternative Command mode disabled. |

**Bit 16 – ACMEN** Alternative Command Mode Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Alternative Command mode enabled. |

**Bit 15 – CLEAR** Bus CLEAR Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If Master mode is enabled, send a bus clear command. |

**Bit 14 – PECRQ** PEC Request

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | A PEC check or transmission is requested. |

**Bit 13 – PECDIS** Packet Error Checking Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | SMBus PEC (CRC) generation and check disabled. |

**Bit 12 – PECEN** Packet Error Checking Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | SMBus PEC (CRC) generation and check enabled. |

**Bit 11 – SMBDIS** SMBus Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | SMBus mode disabled. |

**Bit 10 – SMBEN** SMBus Mode Enabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If SMBDIS = 0, SMBus mode enabled. |

**Bit 9 – HSDIS** TWI High-Speed Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | High-speed mode disabled. |

**Bit 8 – HSEN** TWI High-Speed Mode Enabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | High-speed mode enabled. |

**Bit 7 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Equivalent to a system reset. |

**Bit 6 – QUICK** SMBus Quick Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If Master mode is enabled, a SMBus Quick Command is sent. |

**Bit 5 – SVDIS** TWI Slave Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Slave mode is disabled. The shifter and holding characters (if it contains data) are transmitted in the case of a read operation. In a write operation, the character being transferred must be completely received before disabling. |

**Bit 4 – SVEN** TWI Slave Mode Enabled
Switching from Master to Slave mode is only permitted when TXCOMP = 1.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Slave mode (SVDIS must be written to 0). |

**Bit 3 – MSDIS** TWI Master Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The Master mode is disabled, all pending data is transmitted. The shifter and holding characters (if it contains data) are transmitted in the case of a write operation. In a read operation, the character being transferred must be completely received before disabling. |

**Bit 2 – MSEN** TWI Master Mode Enabled
Switching from Slave to Master mode is only permitted when TXCOMP = 1.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Master mode (MSDIS must be written to 0). |

**Bit 1 – STOP** Send a STOP Condition

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | STOP condition is sent just after completing the current byte transmission in Master Read mode.<br>– In single data byte master read, both START and STOP must be set.<br>– In multiple data bytes master read, the STOP must be set after the last data received but one.<br>– In Master Read mode, if a NACK bit is received, the STOP is automatically performed.<br>– In master data write operation, a STOP condition will be sent after the transmission of the current data is finished. |

**Bit 0 – START** Send a START Condition
This action is necessary when the TWI peripheral needs to read data from a slave. When configured in Master mode with a write operation, a frame is sent as soon as the user writes a character in the Transmit Holding Register (FLEX_TWI_THR).

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | A frame beginning with a START bit is transmitted according to the features defined in the TWI Master Mode Register (FLEX_TWI_MMR). |

### 46.10.62 TWI Master Mode Register

**Name:** FLEX_TWI_MMR
**Offset:** 0x604
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | NOAP |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | DADR[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | MREAD | | | IADRSZ[1:0] | |
| Access | | | | R/W | | | R/W | R/W |
| Reset | | | | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 24 – NOAP** No Auto-Stop On NACK Error

| Value | Description |
|---|---|
| 0 | A stop condition is sent automatically upon Not-Acknowledge error detection. |
| 1 | No automatic action is performed upon Not-Acknowledge error detection. |

**Bits 22:16 – DADR[6:0]** Device Address
The device address is used to access slave devices in Read or Write mode. Those bits are only used in Master mode.

**Bit 12 – MREAD** Master Read Direction

| Value | Description |
|---|---|
| 0 | Master write direction. |
| 1 | Master read direction. |

**Bits 9:8 – IADRSZ[1:0]** Internal Device Address Size

| Value | Name | Description |
|---|---|---|
| 0 | NONE | No internal device address |
| 1 | 1_BYTE | One-byte internal device address |
| 2 | 2_BYTE | Two-byte internal device address |

| Value | Name | Description |
|---|---|---|
| 3 | 3_BYTE | Three-byte internal device address |

### 46.10.63 TWI Slave Mode Register

**Name:** FLEX_TWI_SMR
**Offset:** 0x608
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TWI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATAMEN | SADR3EN | SADR2EN | SADR1EN | | | | |
| Access | R/W | R/W | R/W | R/W | | | | |
| Reset | 0 | 0 | 0 | 0 | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | SADR[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | MASK[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | SCLWSDIS | | SADAT | SMHH | SMDA | | NACKEN |
| Access | | R/W | | R/W | R/W | R/W | | R/W |
| Reset | | 0 | | 0 | 0 | 0 | | 0 |

**Bit 31 – DATAMEN** Data Matching Enable

| Value | Description |
|---|---|
| 0 | Data matching on first received data is disabled. |
| 1 | Data matching on first received data is enabled. |

**Bit 30 – SADR3EN** Slave Address 3 Enable

| Value | Description |
|---|---|
| 0 | Slave address 3 matching is disabled. |
| 1 | Slave address 3 matching is enabled. |

**Bit 29 – SADR2EN** Slave Address 2 Enable

| Value | Description |
|---|---|
| 0 | Slave address 2 matching is disabled. |
| 1 | Slave address 2 matching is enabled. |

**Bit 28 – SADR1EN** Slave Address 1 Enable

| Value | Description |
|---|---|
| 0 | Slave address 1 matching is disabled. |

| Value | Description |
|---|---|
| 1 | Slave address 1 matching is enabled. |

**Bits 22:16 – SADR[6:0]** Slave Address
The slave device address is used in Slave mode in order to be accessed by master devices in Read or Write mode.

SADR must be programmed before enabling the Slave mode or after a general call. Writes at other times have no effect.

**Bits 14:8 – MASK[6:0]** Slave Address Mask
A mask can be applied on the slave device address in Slave mode in order to allow multiple address answer. For each bit of the MASK field set to one, the corresponding SADR bit will be masked.

If the MASK field is set to 0, no mask is applied to the SADR field.

**Bit 6 – SCLWSDIS** Clock Wait State Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clock stretching disabled in Slave mode, OVRE and UNRE will indicate overrun and underrun. |

**Bit 4 – SADAT** Slave Address Treated as Data

| Value | Description |
|---|---|
| 0 | Slave address is handled normally (will not trig RXRDY flag and will not fill FLEX_TWI_RHR upon reception). |
| 1 | Slave address is handled as data field, RXRDY will be set and FLEX_TWI_RHR filled upon slave address reception. |

**Bit 3 – SMHH** SMBus Host Header

| Value | Description |
|---|---|
| 0 | Acknowledge of the SMBus Host Header disabled. |
| 1 | Acknowledge of the SMBus Host Header enabled. |

**Bit 2 – SMDA** SMBus Default Address

| Value | Description |
|---|---|
| 0 | Acknowledge of the SMBus Default Address disabled. |
| 1 | Acknowledge of the SMBus Default Address enabled. |

**Bit 0 – NACKEN** Slave Receiver Data Phase NACK Enable

| Value | Description |
|---|---|
| 0 | Normal value to be returned in the ACK cycle of the data phase in Slave Receiver mode. |
| 1 | NACK value to be returned in the ACK cycle of the data phase in Slave Receiver mode. |

### 46.10.64 TWI Internal Address Register

**Name:** FLEX_TWI_IADR
**Offset:** 0x60C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IADR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IADR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IADR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – IADR[23:0]** Internal Address
0, 1, 2 or 3 bytes depending on IADRSZ.

### 46.10.65 TWI Clock Waveform Generator Register

**Name:** FLEX_TWI_CWGR
**Offset:** 0x610
**Reset:** 0x00000000
**Property:** Read/Write

FLEX_TWI_CWGR is only used in Master mode.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | HOLD[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | BRSRCCLK | | CKDIV[2:0] | | |
| Access | | | | R/W | | R/W | R/W | R/W |
| Reset | | | | 0 | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CHDIV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CLDIV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:24 – HOLD[5:0]** TWD Hold Time Versus TWCK Falling
If High-speed mode is selected TWD is internally modified on the TWCK falling edge to meet the $I^2C$ specified maximum hold time, else if High-speed mode is not configured TWD is kept unchanged after TWCK falling edge for a period of (HOLD + 3) × $t_{peripheral\ clock}$.

**Bit 20 – BRSRCCLK** Bit Rate Source Clock

| Value | Name | Description |
|---|---|---|
| 0 | PERIPH_CLK | The peripheral clock is the source clock for the bit rate generation. |
| 1 | GCLK | GCLK is the source clock for the bit rate generation, thus the bit rate can be independent of the core/peripheral clock. |

**Bits 18:16 – CKDIV[2:0]** Clock Divider
The CKDIV is used to increase both SCL high and low periods.

**Bits 15:8 – CHDIV[7:0]** Clock High Divider
The SCL high period is defined as follows:

If BRSRCCLK = 0: CHDIV = $((t_{high} / t_{peripheral\ clock}) - 3) / 2^{CKDIV}$

If BRSRCCLK = 1: CHDIV = $(t_{high} / t_{ext_ck}) / 2^{CKDIV}$

**Bits 7:0 – CLDIV[7:0]** Clock Low Divider
The SCL low period is defined as follows:

If BRSRCCLK = 0: CLDIV = $((t_{low} / t_{peripheral\ clock}) - 3) / 2^{CKDIV}$

If BRSRCCLK = 1: CLDIV = $(t_{low} / t_{ext_ck}) / 2^{CKDIV}$

### 46.10.66 TWI Status Register

**Name:** FLEX_TWI_SR
**Offset:** 0x620
**Reset:** 0x0300F009
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SR | SDA | SCL |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | LOCK | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| Access | R | | R | R | R | R | | R |
| Reset | 0 | | 0 | 0 | 0 | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOSACC | SCLWS | ARBLST | NACK |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | GACC | SVACC | SVREAD | TXRDY | RXRDY | TXCOMP |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |

**Bit 26 – SR** Start Repeated

| Value | Description |
|---|---|
| 0 | No repeated start has been detected since last FLEX_TWI_SR read. |
| 1 | At least one repeated start has been detected since last FLEX_TWI_SR read. |

**Bit 25 – SDA** SDA Line Value

| Value | Description |
|---|---|
| 0 | SDA line sampled value is ‘0’. |
| 1 | SDA line sampled value is ‘1’. |

**Bit 24 – SCL** SCL Line Value

| Value | Description |
|---|---|
| 0 | SCL line sampled value is ‘0’. |
| 1 | SCL line sampled value is ‘1.’ |

**Bit 23 – LOCK** TWI Lock Due to Frame Errors

| Value | Description |
|---|---|
| 0 | The TWI is not locked. |

| Value | Description |
|---|---|
| 1 | The TWI is locked due to frame errors (see 46.9.3.12 Handling Errors in Alternative Command and 46.9.6 TWI FIFOs). |

**Bit 21 – SMBHHM** SMBus Host Header Address Match (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus Host Header Address received. |
| 1 | A SMBus Host Header Address was received. |

**Bit 20 – SMBDAM** SMBus Default Address Match (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus Default Address received. |
| 1 | A SMBus Default Address was received. |

**Bit 19 – PECERR** PEC Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus PEC error occurred. |
| 1 | A SMBus PEC error occurred. |

**Bit 18 – TOUT** Timeout Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus timeout occurred. |
| 1 | SMBus timeout occurred. |

**Bit 16 – MCACK** Master Code Acknowledge (cleared on read)
MACK used in Slave mode:

| Value | Description |
|---|---|
| 0 | No master code has been received. |
| 1 | A master code has been received. |

**Bit 11 – EOSACC** End Of Slave Access (cleared on read)
This bit is only used in Slave mode.

EOSACC behavior can be seen in figures Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | A slave access is being performing. |
| 1 | The Slave Access is finished. End Of Slave Access is automatically set as soon as SVACC is reset. |

**Bit 10 – SCLWS** Clock Wait State
This bit is only used in Slave mode.

SCLWS behavior can be seen in figures Clock Stretching in Read Mode and Clock Stretching in Write Mode.

| Value | Description |
|---|---|
| 0 | The clock is not stretched. |
| 1 | The clock is stretched. FLEX_TWI_THR / FLEX_TWI_RHR buffer is not filled / emptied before the transmission / reception of a new character. |

**Bit 9 – ARBLST** Arbitration Lost (cleared on read)
This bit is only used in Master mode.

| Value | Description |
|---|---|
| 0 | Arbitration won. |
| 1 | Arbitration lost. Another master of the TWI bus has won the multi-master arbitration. TXCOMP is set at the same time. |

**Bit 8 – NACK** Not Acknowledged (cleared on read)
NACK used in Master mode:

0: Each data byte has been correctly received by the far-end side TWI slave component.

1: A data or address byte has not been acknowledged by the slave component. Set at the same time as TXCOMP.

NACK used in Slave Read mode:

0: Each data byte has been correctly received by the master.

1: In Read mode, a data byte has not been acknowledged by the master. When NACK is set, the user must not fill FLEX_TWI_THR even if TXRDY is set, because it means that the master will stop the data transfer or reinitiate it..

Note that in Slave Write mode, all data are acknowledged by the TWI.

**Bit 7 – UNRE** Underrun Error (cleared on read)
This bit is only used in Slave mode if clock stretching is disabled.

| Value | Description |
|---|---|
| 0 | FLEX_TWI_THR has been filled on time. |
| 1 | FLEX_TWI_THR has not been filled on time. |

**Bit 6 – OVRE** Overrun Error (cleared on read)
This bit is only used in Slave mode if clock stretching is disabled.

| Value | Description |
|---|---|
| 0 | FLEX_TWI_RHR has not been loaded while RXRDY was set. |
| 1 | FLEX_TWI_RHR has been loaded while RXRDY was set. Reset by read in FLEX_TWI_SR when TXCOMP is set. |

**Bit 5 – GACC** General Call Access (cleared on read)
This bit is only used in Slave mode.

GACC behavior can be seen in figure Master Performs a General Call.

| Value | Description |
|---|---|
| 0 | No general call has been detected. |

| Value | Description |
|---|---|
| 1 | A general call has been detected. After the detection of general call, if need be, the user may acknowledge this access and decode the following bytes and respond according to the value of the bytes. |

**Bit 4 – SVACC** Slave Access
This bit is only used in Slave mode.

SVACC behavior can be seen in figures Read Access Ordered by a Master, Write Access Ordered by a Master, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | TWI is not addressed. SVACC is automatically cleared after a NACK or a STOP condition is detected. |
| 1 | Indicates that the address decoding sequence has matched (a master has sent SADR). SVACC remains high until a NACK or a STOP condition is detected. |

**Bit 3 – SVREAD** Slave Read
This bit is only used in Slave mode. When SVACC is low (no slave access has been detected) SVREAD is irrelevant.

SVREAD behavior can be seen in figures Read Access Ordered by a Master, Write Access Ordered by a Master, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | Indicates that a write access is performed by a master. |
| 1 | Indicates that a read access is performed by a master. |

**Bit 2 – TXRDY** Transmit Holding Register Ready (cleared by writing FLEX_TWI_THR)
TXRDY used in Master mode:

0: The transmit holding register has not been transferred into the internal shifter. Set to 0 when writing into FLEX_TWI_THR.

1: As soon as a data byte is transferred from FLEX_TWI_THR to internal shifter or if a NACK error is detected, TXRDY is set at the same time as TXCOMP and NACK. TXRDY is also set when MSEN is set (enables TWI).

TXRDY behavior in Master mode can be seen in figures Master Write with One Data Byte,Master Write with Multiple Data Bytes and Master Write with One Byte Internal Address and Multiple Data Bytes.

TXRDY used in Slave mode:

0: As soon as data is written in FLEX_TWI_THR, until this data has been transmitted and acknowledged (ACK or NACK).

1: Indicates that FLEX_TWI_THR is empty and that data has been transmitted and acknowledged.

If TXRDY is high and if a NACK has been detected, the transmission will be stopped. Thus when TRDY = NACK = 1, the user must not fill FLEX_TWI_THR to avoid losing it.

TXRDY behavior in Slave mode can be seen in figures Read Access Ordered by a Master, Clock Stretching in Read Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

When FIFOs are enabled:

0: Transmit FIFO is full and cannot accept more data.

1: Transmit FIFO is not full; one or more data can be written according to TXRDYM field configuration.

TXRDY behavior with FIFOs enabled is illustrated in TXRDY and RXRDY Behavior.

**Bit 1 – RXRDY** Receive Holding Register Ready (cleared when reading FLEX_TWI_RHR)
When FIFOs are disabled:

0: No character has been received since the last FLEX_TWI_RHR read operation.

1: A byte has been received in FLEX_TWI_RHR since the last read.

RXRDY behavior in Master mode can be seen in figure Master Read with Multiple Data Bytes.

RXRDY behavior in Slave mode can be seen in figures Write Access Ordered by a Master, Clock Stretching in Write Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

When FIFOs are enabled:

0: Receive FIFO is empty; no data to read.

1: At least one unread data is in the Receive FIFO.

RXRDY behavior with FIFO enabled is illustrated in TXRDY and RXRDY Behavior.

**Bit 0 – TXCOMP** Transmission Completed (cleared by writing FLEX_TWI_THR)
TXCOMP used in Master mode:

0: During the length of the current frame.

1: When both the holding register and the internal shifter are empty and STOP condition has been sent.

TXCOMP behavior in Master mode can be seen in figures Master Write with One Byte Internal Address and Multiple Data Bytes and Master Read with Multiple Data Bytes.

TXCOMP used in Slave mode:

0: As soon as a Start is detected.

1: After a Stop or a Repeated Start + an address different from SADR is detected.

TXCOMP behavior in Slave mode can be seen in figures Clock Stretching in Read Mode, Clock Stretching in Write Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

### 46.10.67 TWI Status Register (FIFO ENABLED)

**Name:** FLEX_TWI_SR (FIFO_ENABLED)
**Offset:** 0x620
**Reset:** 0x0300F009
**Property:** Read-only

If FIFO is enabled (FLEX_US_CR.FIFOEN bit), see TWI Multiple Data Mode for details.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SR | SDA | SCL |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TXFLOCK | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| Access | R | | R | R | R | R | | R |
| Reset | 0 | | 0 | 0 | 0 | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOSACC | SCLWS | ARBLST | NACK |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | GACC | SVACC | SVREAD | TXRDY | RXRDY | TXCOMP |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |

**Bit 26 – SR** Start Repeated

| Value | Description |
|---|---|
| 0 | No repeated start has been detected since last FLEX_TWI_SR read. |
| 1 | At least one repeated start has been detected since last FLEX_TWI_SR read. |

**Bit 25 – SDA** SDA Line Value

| Value | Description |
|---|---|
| 0 | SDA line sampled value is ‘0’. |
| 1 | SDA line sampled value is ‘1’. |

**Bit 24 – SCL** SCL Line Value

| Value | Description |
|---|---|
| 0 | SCL line sampled value is ‘0’. |
| 1 | SCL line sampled value is ‘1.’ |

**Bit 23 – TXFLOCK** Transmit FIFO Lock

| Value | Description |
|---|---|
| 0 | The Transmit FIFO is not locked. |

| Value | Description |
|---|---|
| 1 | The Transmit FIFO is locked. |

**Bit 21 – SMBHHM** SMBus Host Header Address Match (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus Host Header Address received. |
| 1 | A SMBus Host Header Address was received. |

**Bit 20 – SMBDAM** SMBus Default Address Match (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus Default Address received. |
| 1 | A SMBus Default Address was received. |

**Bit 19 – PECERR** PEC Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus PEC error occurred. |
| 1 | A SMBus PEC error occurred. |

**Bit 18 – TOUT** Timeout Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus timeout occurred. |
| 1 | SMBus timeout occurred. |

**Bit 16 – MCACK** Master Code Acknowledge (cleared on read)
MACK used in Slave mode:

| Value | Description |
|---|---|
| 0 | No master code has been received. |
| 1 | A master code has been received. |

**Bit 11 – EOSACC** End Of Slave Access (cleared on read)
This bit is only used in Slave mode.

EOSACC behavior can be seen in figures Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | A Slave Access is being performed. |
| 1 | The Slave Access is finished. End Of Slave Access is automatically set as soon as SVACC is reset. |

**Bit 10 – SCLWS** Clock Wait State
This bit is only used in Slave mode.

SCLWS behavior can be seen in figures Clock Stretching in Read Mode and Clock Stretching in Write Mode.

| Value | Description |
|---|---|
| 0 | The clock is not stretched. |

| Value | Description |
|---|---|
| 1 | The clock is stretched. FLEX_TWI_THR / FLEX_TWI_RHR buffer is not filled / emptied before the transmission / reception of a new character. |

**Bit 9 – ARBLST** Arbitration Lost (cleared on read)
This bit is only used in Master mode.

| Value | Description |
|---|---|
| 0 | Arbitration won. |
| 1 | Arbitration lost. Another master of the TWI bus has won the multi-master arbitration. TXCOMP is set at the same time. |

**Bit 8 – NACK** Not Acknowledged (cleared on read)
NACK used in Master mode:

0: Each data byte has been correctly received by the far-end side TWI slave component.

1: A data or address byte has not been acknowledged by the slave component. Set at the same time as TXCOMP.

NACK used in Slave Read mode:

0: Each data byte has been correctly received by the master.

1: In Read mode, a data byte has not been acknowledged by the master. When NACK is set the user must not fill FLEX_TWI_THR even if TXRDY is set, because it means that the master will stop the data transfer or re initiate it.

Note that in Slave Write mode all data are acknowledged by the TWI.

**Bit 7 – UNRE** Underrun Error (cleared on read)
This bit is only used in Slave mode if clock stretching is disabled.

| Value | Description |
|---|---|
| 0 | FLEX_TWI_THR has been filled on time. |
| 1 | FLEX_TWI_THR has not been filled on time. |

**Bit 6 – OVRE** Overrun Error (cleared on read)
This bit is only used in Slave mode if clock stretching is disabled.

| Value | Description |
|---|---|
| 0 | FLEX_TWI_RHR has not been loaded while RXRDY was set. |
| 1 | FLEX_TWI_RHR has been loaded while RXRDY was set. Reset by read in FLEX_TWI_SR when TXCOMP is set. |

**Bit 5 – GACC** General Call Access (cleared on read)
This bit is only used in Slave mode.

GACC behavior can be seen in figure Master Performs a General Call.

| Value | Description |
|---|---|
| 0 | No general call has been detected. |
| 1 | A general call has been detected. After the detection of general call, if need be, the user may acknowledge this access and decode the following bytes and respond according to the value of the bytes. |

**Bit 4 – SVACC** Slave Access
This bit is only used in Slave mode.

SVACC behavior can be seen in figures Read Access Ordered by a Master, Write Access Ordered by a Master, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | TWI is not addressed. SVACC is automatically cleared after a NACK or a STOP condition is detected. |
| 1 | Indicates that the address decoding sequence has matched (a master has sent SADR). SVACC remains high until a NACK or a STOP condition is detected. |

**Bit 3 – SVREAD** Slave Read
This bit is only used in Slave mode. When SVACC is low (no slave access has been detected) SVREAD is irrelevant.

SVREAD behavior can be seen in figures Read Access Ordered by a Master, Write Access Ordered by a Master, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | Indicates that a write access is performed by a master. |
| 1 | Indicates that a read access is performed by a master. |

**Bit 2 – TXRDY** Transmit Holding Register Ready (cleared by writing FLEX_TWI_THR)
TXRDY used in Master mode:

0: The transmit holding register has not been transferred into the internal shifter. Set to 0 when writing into FLEX_TWI_THR.

1: As soon as a data byte is transferred from FLEX_TWI_THR to internal shifter or if a NACK error is detected, TXRDY is set at the same time as TXCOMP and NACK. TXRDY is also set when MSEN is set (enables TWI).

TXRDY behavior in Master mode can be seen in figures Master Write with One Data Byte,Master Write with Multiple Data Bytes and Master Write with One Byte Internal Address and Multiple Data Bytes.

TXRDY used in Slave mode:

0: As soon as data is written in FLEX_TWI_THR, until this data has been transmitted and acknowledged (ACK or NACK).

1: Indicates that FLEX_TWI_THR is empty and that data has been transmitted and acknowledged.

If TXRDY is high and if a NACK has been detected, the transmission will be stopped. Thus when TRDY = NACK = 1, the user must not fill FLEX_TWI_THR to avoid losing it.

TXRDY behavior in Slave mode can be seen in figures Read Access Ordered by a Master, Clock Stretching in Read Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

When FIFOs are enabled:

0: Transmit FIFO is full and cannot accept more data.

1: Transmit FIFO is not full; one or more data can be written according to TXRDYM field configuration.

TXRDY behavior with FIFOs enabled is illustrated in TXRDY and RXRDY Behavior.

**Bit 1 – RXRDY** Receive Holding Register Ready (cleared when reading FLEX_TWI_RHR)
When FIFOs are disabled:

0: No character has been received since the last FLEX_TWI_RHR read operation.

1: A byte has been received in FLEX_TWI_RHR since the last read.

RXRDY behavior in Master mode can be seen in figure Master Read with Multiple Data Bytes.

RXRDY behavior in Slave mode can be seen in figures Write Access Ordered by a Master, Clock Stretching in Write Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

When FIFOs are enabled:

0: Receive FIFO is empty; no data to read.

1: At least one unread data is in the Receive FIFO.

RXRDY behavior with FIFO enabled is illustrated in TXRDY and RXRDY Behavior.

**Bit 0 – TXCOMP** Transmission Completed (cleared by writing FLEX_TWI_THR)
TXCOMP used in Master mode:

0: During the length of the current frame.

1: When both holding register and internal shifter are empty and STOP condition has been sent.

TXCOMP behavior in Master mode can be seen in figures Master Write with One Byte Internal Address and Multiple Data Bytes and Master Read with Multiple Data Bytes.

TXCOMP used in Slave mode:

0: As soon as a Start is detected.

1: After a Stop or a Repeated Start + an address different from SADR is detected.

TXCOMP behavior in Slave mode can be seen in figures Clock Stretching in Read Mode, Clock Stretching in Write Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

### 46.10.68 TWI Interrupt Enable Register

**Name:** FLEX_TWI_IER
**Offset:** 0x624
**Reset:** –
**Property:** Write-only

This register can only be written if the WPITEN bit is cleared in the TWI Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| Access | | | W | W | W | W | | W |
| Reset | | | – | – | – | – | | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TXBUFE | RXBUFF | ENDTX | ENDRX | EOSACC | SCL_WS | ARBLST | NACK |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| Access | W | W | W | W | | W | W | W |
| Reset | – | – | – | – | | – | – | – |

**Bit 21 – SMBHHM** SMBus Host Header Address Match Interrupt Enable

**Bit 20 – SMBDAM** SMBus Default Address Match Interrupt Enable

**Bit 19 – PECERR** PEC Error Interrupt Enable

**Bit 18 – TOUT** Timeout Error Interrupt Enable

**Bit 16 – MCACK** Master Code Acknowledge Interrupt Enable

**Bit 15 – TXBUFE** Transmit Buffer Empty Interrupt Enable

**Bit 14 – RXBUFF** Receive Buffer Full Interrupt Enable

**Bit 13 – ENDTX** End of Transmit Buffer Interrupt Enable

**Bit 12 – ENDRX** End of Receive Buffer Interrupt Enable

**Bit 11 – EOSACC** End Of Slave Access Interrupt Enable

**Bit 10 – SCL_WS** Clock Wait State Interrupt Enable

**Bit 9 – ARBLST** Arbitration Lost Interrupt Enable

**Bit 8 – NACK** Not Acknowledge Interrupt Enable

**Bit 7 – UNRE** Underrun Error Interrupt Enable

**Bit 6 – OVRE** Overrun Error Interrupt Enable

**Bit 5 – GACC** General Call Access Interrupt Enable

**Bit 4 – SVACC** Slave Access Interrupt Enable

**Bit 2 – TXRDY** Transmit Holding Register Ready Interrupt Enable

**Bit 1 – RXRDY** Receive Holding Register Ready Interrupt Enable

**Bit 0 – TXCOMP** Transmission Completed Interrupt Enable

### 46.10.69 TWI Interrupt Disable Register

**Name:** FLEX_TWI_IDR
**Offset:** 0x628
**Reset:** –
**Property:** Write-only

This register can only be written if the WPITEN bit is cleared in the TWI Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| Access | | | W | W | W | W | | W |
| Reset | | | – | – | – | – | | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TXBUFE | RXBUFF | ENDTX | ENDRX | EOSACC | SCL_WS | ARBLST | NACK |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| Access | W | W | W | W | | W | W | W |
| Reset | – | – | – | – | | – | – | – |

**Bit 21 – SMBHHM** SMBus Host Header Address Match Interrupt Disable

**Bit 20 – SMBDAM** SMBus Default Address Match Interrupt Disable

**Bit 19 – PECERR** PEC Error Interrupt Disable

**Bit 18 – TOUT** Timeout Error Interrupt Disable

**Bit 16 – MCACK** Master Code Acknowledge Interrupt Disable

**Bit 15 – TXBUFE** Transmit Buffer Empty Interrupt Disable

**Bit 14 – RXBUFF** Receive Buffer Full Interrupt Disable

**Bit 13 – ENDTX** End of Transmit Buffer Interrupt Disable

**Bit 12 – ENDRX** End of Receive Buffer Interrupt Disable

**Bit 11 – EOSACC** End Of Slave Access Interrupt Disable

**Bit 10 – SCL_WS** Clock Wait State Interrupt Disable

**Bit 9 – ARBLST** Arbitration Lost Interrupt Disable

**Bit 8 – NACK** Not Acknowledge Interrupt Disable

**Bit 7 – UNRE** Underrun Error Interrupt Disable

**Bit 6 – OVRE** Overrun Error Interrupt Disable

**Bit 5 – GACC** General Call Access Interrupt Disable

**Bit 4 – SVACC** Slave Access Interrupt Disable

**Bit 2 – TXRDY** Transmit Holding Register Ready Interrupt Disable

**Bit 1 – RXRDY** Receive Holding Register Ready Interrupt Disable

**Bit 0 – TXCOMP** Transmission Completed Interrupt Disable

### 46.10.70 TWI Interrupt Mask Register

**Name:** FLEX_TWI_IMR
**Offset:** 0x62C
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| Access | | | R | R | R | R | | R |
| Reset | | | 0 | 0 | 0 | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TXBUFE | RXBUFF | ENDTX | ENDRX | EOSACC | SCL_WS | ARBLST | NACK |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| Access | R | R | R | R | | R | R | R |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bit 21 – SMBHHM** SMBus Host Header Address Match Interrupt Mask

**Bit 20 – SMBDAM** SMBus Default Address Match Interrupt Mask

**Bit 19 – PECERR** PEC Error Interrupt Mask

**Bit 18 – TOUT** Timeout Error Interrupt Mask

**Bit 16 – MCACK** Master Code Acknowledge Interrupt Mask

**Bit 15 – TXBUFE** Transmit Buffer Empty Interrupt Mask

**Bit 14 – RXBUFF** Receive Buffer Full Interrupt Mask

**Bit 13 – ENDTX** End of Transmit Buffer Interrupt Mask

**Bit 12 – ENDRX** End of Receive Buffer Interrupt Mask

**Bit 11 – EOSACC** End Of Slave Access Interrupt Mask

**Bit 10 – SCL_WS** Clock Wait State Interrupt Mask

**Bit 9 – ARBLST** Arbitration Lost Interrupt Mask

**Bit 8 – NACK** Not Acknowledge Interrupt Mask

**Bit 7 – UNRE** Underrun Error Interrupt Mask

**Bit 6 – OVRE** Overrun Error Interrupt Mask

**Bit 5 – GACC** General Call Access Interrupt Mask

**Bit 4 – SVACC** Slave Access Interrupt Mask

**Bit 2 – TXRDY** Transmit Holding Register Ready Interrupt Mask

**Bit 1 – RXRDY** Receive Holding Register Ready Interrupt Mask

**Bit 0 – TXCOMP** Transmission Completed Interrupt Mask

### 46.10.71 TWI Receive Holding Register

**Name:** FLEX_TWI_RHR
**Offset:** 0x630
**Reset:** 0x00000000
**Property:** Read-only

If FIFO is enabled (FLEX_TWI_CR.FIFOEN bit) and FLEX_TWI_FMR.RXRDYM = 0, see TWI Single Data Mode for details.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXDATA[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – RXDATA[7:0]** Master or Slave Receive Holding Data

### 46.10.72 TWI Receive Holding Register (FIFO Enabled)

**Name:** FLEX_TWI_RHR (FIFO_ENABLED)
**Offset:** 0x630
**Reset:** 0x00000000
**Property:** Read-only

If FIFO is enabled (FLEX_TWI_CR.FIFOEN bit) and FLEX_TWI_FMR.RXRDYM > 0, see TWI Multiple Data Mode for details.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RXDATA3[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RXDATA2[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RXDATA1[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXDATA0[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – RXDATA3[7:0]** Master or Slave Receive Holding Data 3

**Bits 23:16 – RXDATA2[7:0]** Master or Slave Receive Holding Data 2

**Bits 15:8 – RXDATA1[7:0]** Master or Slave Receive Holding Data 1

**Bits 7:0 – RXDATA0[7:0]** Master or Slave Receive Holding Data 0

### 46.10.73 TWI Transmit Holding Register

**Name:** FLEX_TWI_THR
**Offset:** 0x634
**Reset:** –
**Property:** Write-only

If FIFO is enabled (FIFOEN bit in FLEX_TWI_CR) and FLEX_TWI_FMR.TXRDYM = 0, see TWI Single Data Mode for details.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXDATA[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 7:0 – TXDATA[7:0]** Master or Slave Transmit Holding Data

### 46.10.74 TWI Transmit Holding Register (FIFO Enabled)

**Name:** FLEX_TWI_THR (FIFO_ENABLED)
**Offset:** 0x634
**Reset:** –
**Property:** Write-only

If FIFO is enabled (FLEX_US_CR.FIFOEN bit) and FLEX_TWI_FMR.TXRDYM > 0, see TWI Multiple Data Mode for details.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TXDATA3[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TXDATA2[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TXDATA1[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXDATA0[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 31:24 – TXDATA3[7:0]** Master or Slave Transmit Holding Data 3

**Bits 23:16 – TXDATA2[7:0]** Master or Slave Transmit Holding Data 2

**Bits 15:8 – TXDATA1[7:0]** Master or Slave Transmit Holding Data 1

**Bits 7:0 – TXDATA0[7:0]** Master or Slave Transmit Holding Data 0

### 46.10.75 TWI SMBus Timing Register

**Name:** FLEX_TWI_SMBTR
**Offset:** 0x638
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | THMAX[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TLOWM[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TLOWS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PRESC[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 31:24 – THMAX[7:0]** Clock High Maximum Cycles
Clock cycles in clock high maximum count. Prescaled by PRESC. Used for bus free detection. Used to time THIGH:MAX.

**Bits 23:16 – TLOWM[7:0]** Master Clock Stretch Maximum Cycles

| Value | Description |
|---|---|
| 0 | TLOW:MEXT timeout check disabled. |
| 1-255 | Clock cycles in master maximum clock stretch count. Prescaled by PRESC. Used to time TLOW:MEXT. |

**Bits 15:8 – TLOWS[7:0]** Slave Clock Stretch Maximum Cycles

| Value | Description |
|---|---|
| 0 | TLOW:SEXT timeout check disabled. |
| 1-255 | Clock cycles in slave maximum clock stretch count. Prescaled by PRESC. Used to time TLOW:SEXT. |

**Bits 3:0 – PRESC[3:0]** SMBus Clock Prescaler
Used to specify how to prescale the TLOWS, TLOWM and THMAX counters in SMBTR. Counters are prescaled according to the following formula: PRESC = Log(fMCK / $f_{Prescaled}$) / Log(2) - 1

### 46.10.76 TWI Alternative Command Register

**Name:** FLEX_TWI_ACR
**Offset:** 0x640
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | NPEC | NDIR |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NDATAL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | PEC | DIR |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATAL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 25 – NPEC** Next PEC Request (SMBus Mode only)

| Value | Description |
|---|---|
| 0 | The next transfer does not use a PEC byte. |
| 1 | The next transfer uses a PEC byte. |

**Bit 24 – NDIR** Next Transfer Direction

| Value | Description |
|---|---|
| 0 | Write direction. |
| 1 | Read direction. |

**Bits 23:16 – NDATAL[7:0]** Next Data Length

| Value | Description |
|---|---|
| 0 | No data to send (see 46.9.3.11 Alternative Command). |
| 1-255 | Number of bytes to send for the next transfer. |

**Bit 9 – PEC** PEC Request (SMBus Mode only)

| Value | Description |
|---|---|
| 0 | The transfer does not use a PEC byte. |
| 1 | The transfer uses a PEC byte. |

**Bit 8 – DIR** Transfer Direction

| Value | Description |
|---|---|
| 0 | Write direction. |
| 1 | Read direction. |

**Bits 7:0 – DATAL[7:0]** Data Length

| Value | Description |
|---|---|
| 0 | No data to send (see 46.9.3.11 Alternative Command). |
| 1-255 | Number of bytes to send during the transfer. |

### 46.10.77 TWI Filter Register

**Name:** FLEX_TWI_FILTR
**Offset:** 0x644
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | THRES[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PADFCFG | PADFEN | FILT |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 10:8 – THRES[2:0]** Digital Filter Threshold

| Value | Description |
|---|---|
| 0 | No filtering applied on TWI inputs. |
| 1-7 | Maximum pulse width of spikes which will be suppressed by the input filter, defined in peripheral clock cycles. |

**Bit 2 – PADFCFG** PAD Filter Config
Refer to the section “Electrical Characteristics” for filter configuration details.

**Bit 1 – PADFEN** PAD Filter Enable

| Value | Description |
|---|---|
| 0 | PAD analog filter is disabled. |
| 1 | PAD analog filter is enabled. (The analog filter must be enabled if High-speed mode is enabled.) |

**Bit 0 – FILT** RX Digital Filter
TWI digital input filtering follows a majority decision based on three samples from SDA/SCL lines at peripheral clock frequency.

| Value | Description |
|---|---|
| 0 | No filtering applied on TWI inputs. |
| 1 | TWI input filtering is active. (Only in Standard and Fast modes) |

### 46.10.78 TWI SleepWalking Matching Register

**Name:** FLEX_TWI_SWMR
**Offset:** 0x64C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATAM[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | SADR3[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | SADR2[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | SADR1[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – DATAM[7:0]** Data Match
The TWI module will extend the SleepWalking matching process to the first received data comparing it with DATAM if DATAMEN bit is enabled.

**Bits 22:16 – SADR3[6:0]** Slave Address 3
Slave address 3. The TWI module will match on this additional address if SADR3EN bit is enabled.

**Bits 14:8 – SADR2[6:0]** Slave Address 2
Slave address 2. The TWI module will match on this additional address if SADR2EN bit is enabled.

**Bits 6:0 – SADR1[6:0]** Slave Address 1
Slave address 1. The TWI module will match on this additional address if SADR1EN bit is enabled.

### 46.10.79 TWI FIFO Mode Register

**Name:** FLEX_TWI_FMR
**Offset:** 0x650
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TWI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | RXFTHRES[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | TXFTHRES[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | RXRDYM[1:0] | | | | TXRDYM[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bits 29:24 – RXFTHRES[5:0]** Receive FIFO Threshold

| Value | Description |
|---|---|
| 0-16 | Defines the Receive FIFO threshold value (number of data). The FLEX_TWI_FSR.RXFTH flag will be set when Receive FIFO goes from “below” threshold state to “equal to or above” threshold state. |

**Bits 21:16 – TXFTHRES[5:0]** Transmit FIFO Threshold

| Value | Description |
|---|---|
| 0-16 | Defines the Transmit FIFO threshold value (number of data). The FLEX_TWI_FSR.TXFTH flag will be set when Transmit FIFO goes from “above” threshold state to “equal to or below” threshold state. |

**Bits 5:4 – RXRDYM[1:0]** Receiver Ready Mode
If FIFOs are enabled, the FLEX_TWI_SR.RXRDY flag behaves as follows.

| Value | Name | Description |
|---|---|---|
| 0 | ONE_DATA | RXRDY will be at level ‘1’ when at least one unread data is in the Receive FIFO |

| Value | Name | Description |
|---|---|---|
| 1 | TWO_DATA | RXRDY will be at level ‘1’ when at least two unread data are in the Receive FIFO |
| 2 | FOUR_DATA | RXRDY will be at level ‘1’ when at least four unread data are in the Receive FIFO |

**Bits 1:0 – TXRDYM[1:0]** Transmitter Ready Mode
If FIFOs are enabled, the FLEX_TWI_SR.TXRDY flag behaves as follows.

| Value | Name | Description |
|---|---|---|
| 0 | ONE_DATA | TXRDY will be at level ‘1’ when at least one data can be written in the Transmit FIFO |
| 1 | TWO_DATA | TXRDY will be at level ‘1’ when at least two data can be written in the Transmit FIFO |
| 2 | FOUR_DATA | TXRDY will be at level ‘1’ when at least four data can be written in the Transmit FIFO |

### 46.10.80 TWI FIFO Level Register

**Name:** FLEX_TWI_FLR
**Offset:** 0x654
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | RXFL[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TXFL[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 21:16 – RXFL[5:0]** Receive FIFO Level

| Value | Description |
|---|---|
| 0 | There is no unread data in the Receive FIFO. |
| 1-16 | Indicates the number of unread data in the Receive FIFO. |

**Bits 5:0 – TXFL[5:0]** Transmit FIFO Level

| Value | Description |
|---|---|
| 0 | There is no data in the Transmit FIFO. |
| 1-16 | Indicates the number of data in the Transmit FIFO. |

### 46.10.81 TWI FIFO Status Register

**Name:** FLEX_TWI_FSR
**Offset:** 0x660
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – RXFPTEF** Receive FIFO Pointer Error Flag
See 46.9.6.10 FIFO Pointer Error for details.

| Value | Description |
|---|---|
| 0 | No Receive FIFO pointer occurred. |
| 1 | Receive FIFO pointer error occurred. Receiver must be reset. |

**Bit 6 – TXFPTEF** Transmit FIFO Pointer Error Flag
See 46.9.6.10 FIFO Pointer Error for details.

| Value | Description |
|---|---|
| 0 | No Transmit FIFO pointer occurred. |
| 1 | Transmit FIFO pointer error occurred. Transceiver must be reset. |

**Bit 5 – RXFTHF** Receive FIFO Threshold Flag

| Value | Description |
|---|---|
| 0 | Number of unread data in Receive FIFO is below RXFTHRES threshold. |
| 1 | Number of unread data in Receive FIFO has reached RXFTHRES threshold since the last read of FLEX_TWI_FSR. |

**Bit 4 – RXFFF** Receive FIFO Full Flag

| Value | Description |
|---|---|
| 0 | Receive FIFO is not empty. |
| 1 | Receive FIFO has been filled since the last read of FLEX_TWI_FSR. |

**Bit 3 – RXFEF** Receive FIFO Empty Flag

| Value | Description |
|---|---|
| 0 | Receive FIFO is not empty. |
| 1 | Receive FIFO has been emptied since the last read of FLEX_TWI_FSR. |

**Bit 2 – TXFTHF** Transmit FIFO Threshold Flag (cleared on read)

| Value | Description |
|---|---|
| 0 | Number of data in Transmit FIFO is above TXFTHRES threshold. |
| 1 | Number of data in Transmit FIFO has reached TXFTHRES threshold since the last read of FLEX_TWI_FSR. |

**Bit 1 – TXFFF** Transmit FIFO Full Flag (cleared on read)

| Value | Description |
|---|---|
| 0 | Transmit FIFO is not full. |
| 1 | Transmit FIFO has been filled since the last read of FLEX_TWI_FSR. |

**Bit 0 – TXFEF** Transmit FIFO Empty Flag (cleared on read)

| Value | Description |
|---|---|
| 0 | Transmit FIFO is not empty. |
| 1 | Transmit FIFO has been emptied since the last read of FLEX_TWI_FSR. |

### 46.10.82 TWI FIFO Interrupt Enable Register

**Name:** FLEX_TWI_FIER
**Offset:** 0x664
**Reset:** –
**Property:** Write-only

This register can only be written if the WPITEN bit is cleared in the TWI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 7 – RXFPTEF** RXFPTEF Interrupt Enable

**Bit 6 – TXFPTEF** TXFPTEF Interrupt Enable

**Bit 5 – RXFTHF** RXFTHF Interrupt Enable

**Bit 4 – RXFFF** RXFFF Interrupt Enable

**Bit 3 – RXFEF** RXFEF Interrupt Enable

**Bit 2 – TXFTHF** TXFTHF Interrupt Enable

**Bit 1 – TXFFF** TXFFF Interrupt Enable

**Bit 0 – TXFEF** TXFEF Interrupt Enable

### 46.10.83 TWI FIFO Interrupt Disable Register

**Name:** FLEX_TWI_FIDR
**Offset:** 0x668
**Reset:** –
**Property:** Write-only

This register can only be written if the WPITEN bit is cleared in the TWI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 7 – RXFPTEF** RXFPTEF Interrupt Disable

**Bit 6 – TXFPTEF** TXFPTEF Interrupt Disable

**Bit 5 – RXFTHF** RXFTHF Interrupt Disable

**Bit 4 – RXFFF** RXFFF Interrupt Disable

**Bit 3 – RXFEF** RXFEF Interrupt Disable

**Bit 2 – TXFTHF** TXFTHF Interrupt Disable

**Bit 1 – TXFFF** TXFFF Interrupt Disable

**Bit 0 – TXFEF** TXFEF Interrupt Disable

### 46.10.84 TWI FIFO Interrupt Mask Register

**Name:** FLEX_TWI_FIMR
**Offset:** 0x66C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – RXFPTEF** RXFPTEF Interrupt Mask

**Bit 6 – TXFPTEF** TXFPTEF Interrupt Mask

**Bit 5 – RXFTHF** RXFTHF Interrupt Mask

**Bit 4 – RXFFF** RXFFF Interrupt Mask

**Bit 3 – RXFEF** RXFEF Interrupt Mask

**Bit 2 – TXFTHF** TXFTHF Interrupt Mask

**Bit 1 – TXFFF** TXFFF Interrupt Mask

**Bit 0 – TXFEF** TXFEF Interrupt Mask

### 46.10.85 TWI Write Protection Mode Register

**Name:** FLEX_TWI_WPMR
**Offset:** 0x6E4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | WPCREN | WPITEN | WPEN |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x545749 | PASSWD | Writing any other value in this field aborts the write operation of bits WPEN, WPITEN and WPCREN.<br>Always reads as 0. |

**Bit 2 – WPCREN** Write Protection Control Enable

| Value | Description |
|---|---|
| 0 | Disables the write protection on control register if WPKEY corresponds to 0x545749 (“TWI” in ASCII). |
| 1 | Enables the write protection on control register if WPKEY corresponds to 0x545749 (“TWI” in ASCII). |

**Bit 1 – WPITEN** Write Protection Interrupt Enable

| Value | Description |
|---|---|
| 0 | Disables the write protection on interrupt registers if WPKEY corresponds to 0x545749 (“TWI” in ASCII). |
| 1 | Enables the write protection on interrupt registers if WPKEY corresponds to 0x545749 (“TWI” in ASCII). |

**Bit 0 – WPEN** Write Protection Enable
See TWI Register Write Protection for the list of registers that can be write-protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x545749 (“TWI” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x545749 (“TWI” in ASCII). |

### 46.10.86 TWI Write Protection Status Register

**Name:** FLEX_TWI_WPSR
**Offset:** 0x6E8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPVSRC[23:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protect Violation Status

| Value | Description |
|---|---|
| 0 | No Write Protection Violation has occurred since the last read of FLEX_TWI_WPSR. |
| 1 | A Write Protection Violation has occurred since the last read of FLEX_TWI_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 47. Universal Asynchronous Receiver Transmitter (UART)

## 47.1 Description

The Universal Asynchronous Receiver Transmitter (UART) features a two-pin UART that can be used for communication and trace purposes and offers an ideal medium for in-situ programming solutions.

Moreover, the association with a DMA controller permits packet handling for these tasks with processor time reduced to a minimum.

## 47.2 Embedded Characteristics

- Two-pin UART
  - Independent Receiver and Transmitter with a Common Programmable Baud Rate Generator
  - Baud Rate can be Driven by Processor-Independent Generic Source Clock
  - Even, Odd, Mark or Space Parity Generation
  - Parity, Framing and Overrun Error Detection
  - Automatic Echo, Local Loopback and Remote Loopback Channel Modes
  - Digital Filter on Receive Line
  - Interrupt Generation
  - Support for Two DMA Channels with Connection to Receiver and Transmitter
  - Supports Asynchronous Partial Wake-up on Receive Line Activity (SleepWalking)
  - Comparison Function on Received Character
  - Receiver Time-out
  - Register Write Protection

## 47.3 Block Diagram

**Figure 47-1. UART Block Diagram**

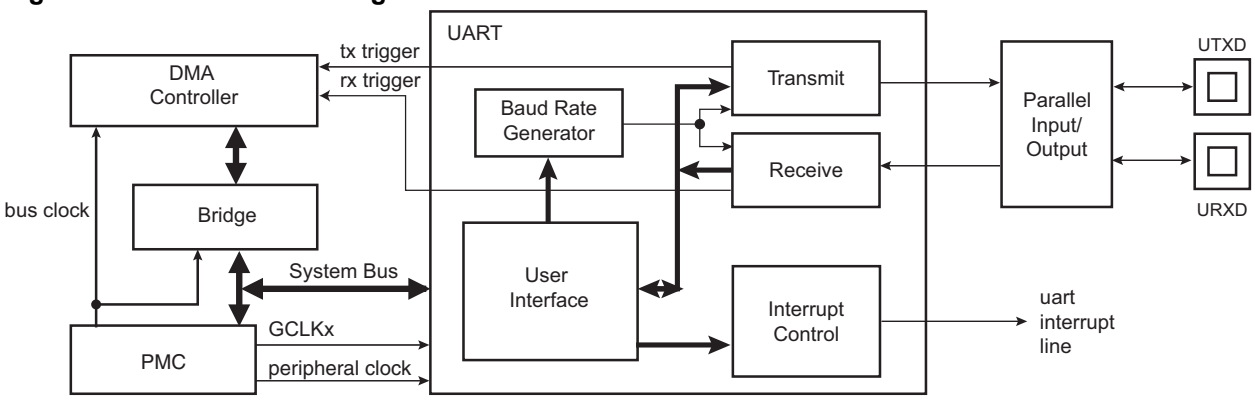



**Table 47-1. UART Pin Description**

| Pin Name | Description | Type |
|---|---|---|
| URXD | UART Receive Data | Input |
| UTXD | UART Trasnmit Data | Output |

## 47.4 Product Dependencies

### 47.4.1 I/O Lines

The UART pins are multiplexed with PIO lines. The user must first configure the corresponding PIO Controller to enable I/O line operations of the UART.

### 47.4.2 Power Management

The UART clock can be controlled through the Power Management Controller (PMC). In this case, the user must first configure the PMC to enable the UART clock. Usually, the peripheral identifier used for this purpose is 1.

In SleepWalking mode (asynchronous partial wake-up), the PMC must be configured to enable SleepWalking for the UART in the Sleepwalking Enable Register (PMC_SLPWK_ER). Depending on the instructions (requests) provided by the UART to the PMC, the system clock may or may not be automatically provided to the UART.

### 47.4.3 Interrupt Sources

The UART interrupt line is connected to one of the interrupt sources of the Interrupt Controller. Interrupt handling requires programming of the Interrupt Controller before configuring the UART.

## 47.5 Functional Description

The UART operates in Asynchronous mode only and supports only 8-bit character handling (with parity). It has no clock pin.

The UART is made up of a receiver and a transmitter that operate independently, and a common baud rate generator. Receiver timeout and transmitter time guard are not implemented. However, all the implemented features are compatible with those of a standard USART.

### 47.5.1 Baud Rate Generator

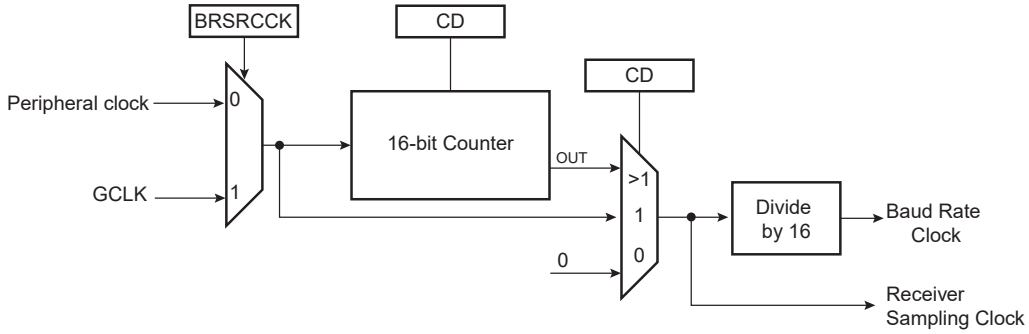
The baud rate generator provides the bit period clock named baud rate clock to both the receiver and the transmitter.

The baud rate clock is the peripheral clock divided by 16 times the clock divisor (CD) value written in the Baud Rate Generator register (UART_BRGR). If UART_BRGR is set to 0, the baud rate clock is disabled and the UART remains inactive. The maximum allowable baud rate is peripheral clock or GCLK divided by 16. The minimum allowable baud rate is peripheral clock divided by (16 x 65536). The clock source driving the baud rate generator (peripheral clock or GCLK) can be selected by writing the bit BRSRCCK in UART_MR.

If GCLK is selected, the baud rate is independent of the processor/bus clock. Thus the processor clock can be changed while UART is enabled. The processor clock frequency changes must be performed only by programming the field PRES in PMC_MCKR (refer to "Power Management Controller (PMC)"). Other methods to modify the processor/bus clock frequency (PLL multiplier, etc.) are forbidden when UART is enabled.

The peripheral clock frequency must be at least three times higher than GCLK.

**Figure 47-2. Baud Rate Generator**




### 47.5.2 Receiver

#### 47.5.2.1 Receiver Reset, Enable and Disable

After device reset, the UART receiver is disabled and must be enabled before being used. The receiver can be enabled by writing the Control Register (UART_CR) with the bit RXEN at 1. At this command, the receiver starts looking for a start bit.

The programmer can disable the receiver by writing UART_CR with the bit RXDIS at 1. If the receiver is waiting for a start bit, it is immediately stopped. However, if the receiver has already detected a start bit and is receiving the data, it waits for the stop bit before actually stopping its operation.

The receiver can be put in reset state by writing UART_CR with the bit RSTRX at 1. In this case, the receiver immediately stops its current operations and is disabled, whatever its current state. If RSTRX is applied when data is being processed, this data is lost.

#### 47.5.2.2 Start Detection and Data Sampling

The UART only supports asynchronous operations, and this affects only its receiver. The UART receiver detects the start of a received character by sampling the URXD signal until it detects a valid start bit. A low level (space) on URXD is interpreted as a valid start bit if it is detected for more than seven cycles of the sampling clock, which is 16 times the baud rate. Hence, a space that is longer than 7/16 of the bit period is detected as a valid start bit. A space which is 7/16 of a bit period or shorter is ignored and the receiver continues to wait for a valid start bit.

When a valid start bit has been detected, the receiver samples the URXD at the theoretical midpoint of each bit. It is assumed that each bit lasts 16 cycles of the sampling clock (1-bit period) so the bit sampling point is eight cycles (0.5-bit period) after the start of the bit. The first sampling point is therefore 24 cycles (1.5-bit periods) after detecting the falling edge of the start bit.

Each subsequent bit is sampled 16 cycles (1-bit period) after the previous one.

**Figure 47-3. Start Bit Detection**

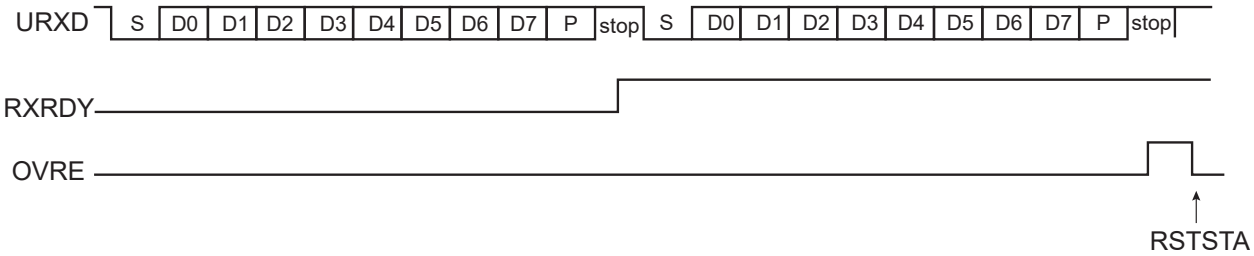
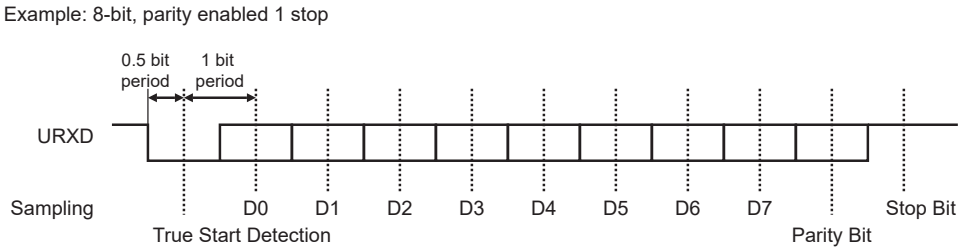


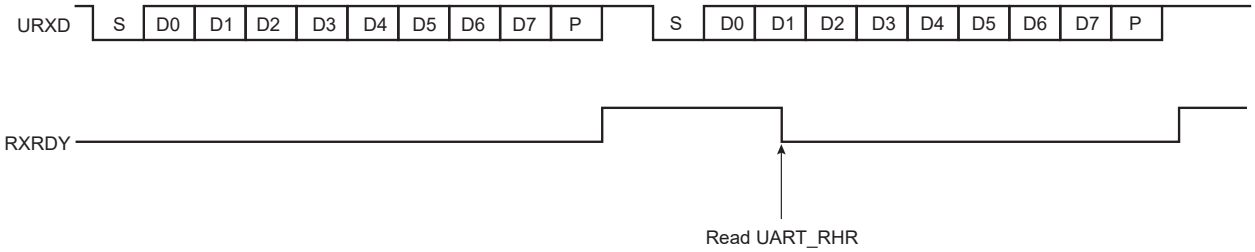
**Figure 47-4. Character Reception**




#### 47.5.2.3 Receiver Ready

When a complete character is received, it is transferred to the Receive Holding Register (UART_RHR) and the RXRDY status bit in the Status Register (UART_SR) is set. The bit RXRDY is automatically cleared when UART_RHR is read.

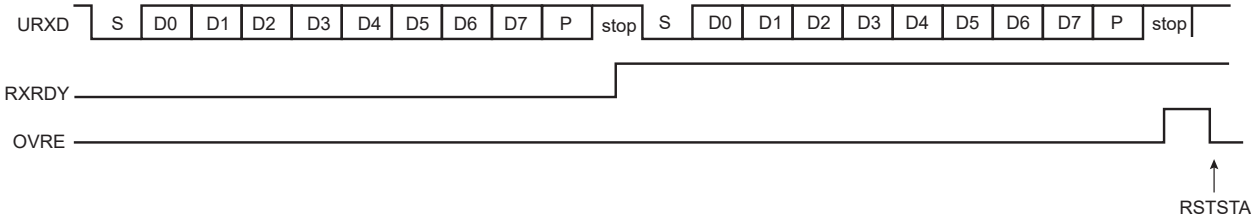
**Figure 47-5. Receiver Ready**




#### 47.5.2.4 Receiver Overrun

The OVRE status bit in UART_SR is set if UART_RHR has not been read by the software (or the DMA Controller) since the last transfer, the RXRDY bit is still set and a new character is received. OVRE is cleared when the software writes a 1 to the bit RSTSTA (Reset Status) in UART_CR.

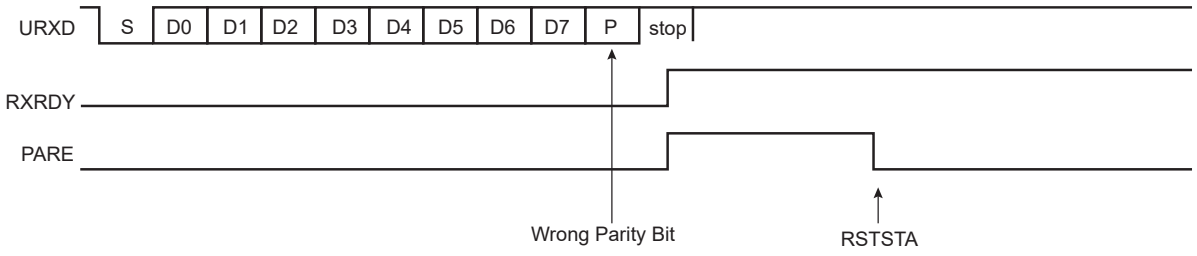
**Figure 47-6. Receiver Overrun**




#### 47.5.2.5 Parity Error

Each time a character is received, the receiver calculates the parity of the received data bits, in accordance with the field PAR in the Mode Register (UART_MR). It then compares the result with the received parity bit. If different, the parity error bit PARE in UART_SR is set at the same time RXRDY is set. The parity bit is cleared when UART_CR is written with the bit RSTSTA (Reset Status) at 1. If a new character is received before the reset status command is written, the PARE bit remains at 1.

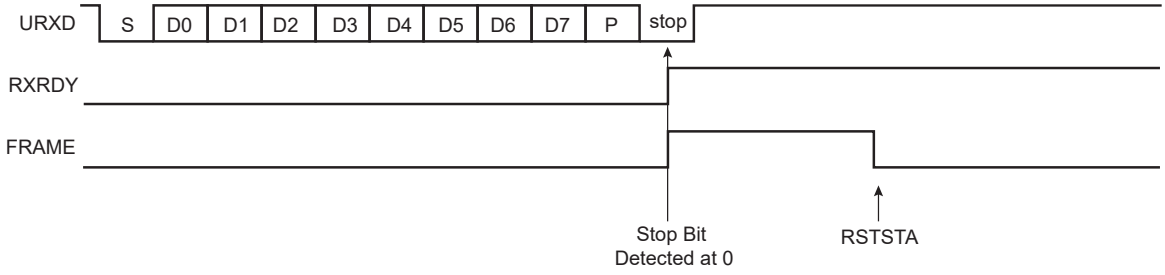
**Figure 47-7. Parity Error**




#### 47.5.2.6 Receiver Framing Error

When a start bit is detected, it generates a character reception when all the data bits have been sampled. The stop bit is also sampled and when it is detected at 0, the FRAME (Framing Error) bit in UART_SR is set at the same time the RXRDY bit is set. The FRAME bit remains high until the Control Register (UART_CR) is written with the bit RSTSTA at 1.

**Figure 47-8. Receiver Framing Error**




#### 47.5.2.7 Receiver Digital Filter

The UART embeds a digital filter on the receive line. It is disabled by default and can be enabled by writing a logical 1 in the FILTER bit of UART_MR. When enabled, the receive line is sampled using the 16x bit clock and a three-sample filter (majority 2 over 3) determines the value of the line.

#### 47.5.2.8 Receiver Time-out

The Receiver Time-out provides support in handling variable-length frames. This feature detects an idle condition on the URXD line. When a time-out is detected, the bit TIMEOUT in the UART_SR rises and can generate an interrupt, thus indicating to the driver an end of frame.

The time-out delay period (during which the receiver waits for a new character) is programmed in the TO field of the Receiver Time-out register (UART_RTOR). If the TO field is written to 0, the Receiver Time-out is disabled and no time-out is detected. The TIMEOUT bit in the UART_SR remains at 0. Otherwise, the receiver loads an 8-bit counter with the value programmed in TO. This counter is decremented at each bit period and reloaded each time a new character is received. If the counter reaches 0, the TIMEOUT bit UART_SR rises. Then, the user can either:

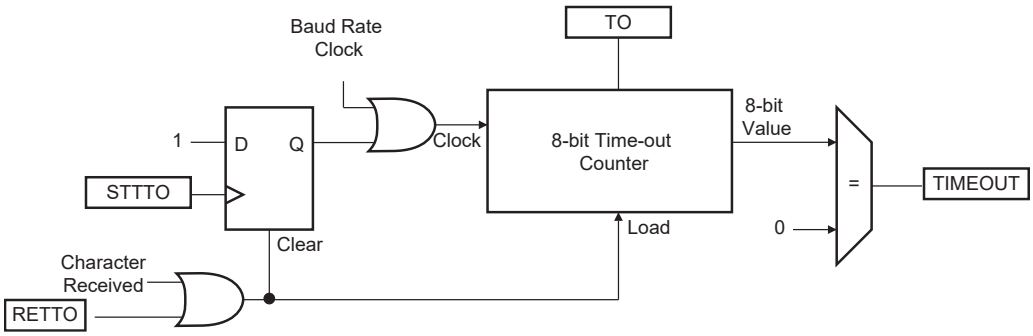
- stop the counter clock until a new character is received. This is performed by writing a one to the STTTO (start Time-out) bit in the UART_CR. In this case, the idle state on URXD before a new character is received does not provide a time-out. This prevents having to handle an interrupt before a character is received and allows waiting for the next idle state on URXD after a frame is received, or
- obtain an interrupt while no character is received. This is performed by writing a one to the RETTO (Reload and Start Time-out) bit in the UART_CR. If RETTO is performed, the counter starts counting down immediately from the TO value. This enables generation of a periodic interrupt so that a user time-out can be handled, for example when no key is pressed on a keyboard.

If STTTO is performed, the counter clock is stopped until a first character is received. The idle state on URXD before the start of the frame does not provide a time-out. This prevents having to obtain a periodic interrupt and enables a wait of the end of frame when the idle state on URXD is detected.

If RETTO is performed, the counter starts counting down immediately from the TO value. This enables generation of a periodic interrupt so that a user time-out can be handled, for example when no key is pressed on a keyboard.

The figure below shows the block diagram of the Receiver Time-out feature.

**Figure 47-9. Receiver Time-out Block Diagram**




The table below gives the maximum time-out period for some standard baud rates.

**Table 47-2. Maximum Time-out Period**

| Baud Rate (bit/s) | Bit Time (µs) | Time-out (µs) |
|---|---|---|
| 600 | 1,667 | 425,085 |
| 1,200 | 833 | 212,415 |
| 2,400 | 417 | 106,335 |
| 4,800 | 208 | 53,040 |
| 9,600 | 104 | 26,520 |
| 14,400 | 69 | 17,595 |
| 19,200 | 52 | 13,260 |
| 28,800 | 35 | 8,925 |
| 38,400 | 26 | 6,630 |
| 56,000 | 18 | 4,590 |
| 57,600 | 17 | 4,335 |
| 200,000 | 5 | 1,275 |

### 47.5.3 Transmitter

#### 47.5.3.1 Transmitter Reset, Enable and Disable

After device reset, the UART transmitter is disabled and must be enabled before being used. The transmitter is enabled by writing UART_CR with the bit TXEN at 1. From this command, the transmitter waits for a character to be written in the Transmit Holding Register (UART_THR) before actually starting the transmission.

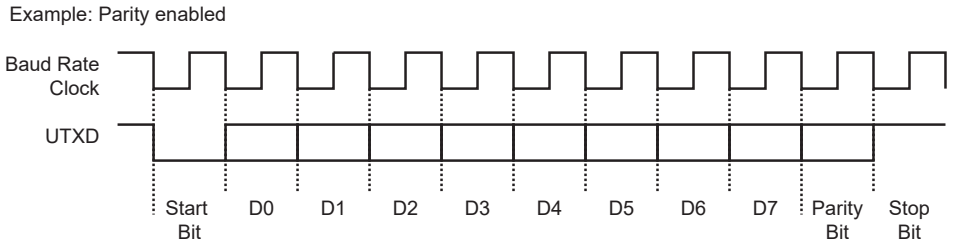
The programmer can disable the transmitter by writing UART_CR with the bit TXDIS at 1. If the transmitter is not operating, it is immediately stopped. However, if a character is being processed into the internal shift register and/or a character has been written in the UART_THR, the characters are completed before the transmitter is actually stopped.

The programmer can also put the transmitter in its reset state by writing the UART_CR with the bit RSTTX at 1. This immediately stops the transmitter, whether or not it is processing characters.

#### 47.5.3.2 Transmit Format

The UART transmitter drives the pin UTXD at the baud rate clock speed. The line is driven depending on the format defined in UART_MR and the data stored in the internal shift register. One start bit at level 0, then the 8 data bits, from the lowest to the highest bit, one optional parity bit and one stop bit at 1 are consecutively shifted out as shown in the following figure. The field PARE in UART_MR defines whether or not a parity bit is shifted out. When a parity bit is enabled, it can be selected between an odd parity, an even parity, or a fixed space or mark bit.

**Figure 47-10. Character Transmission**

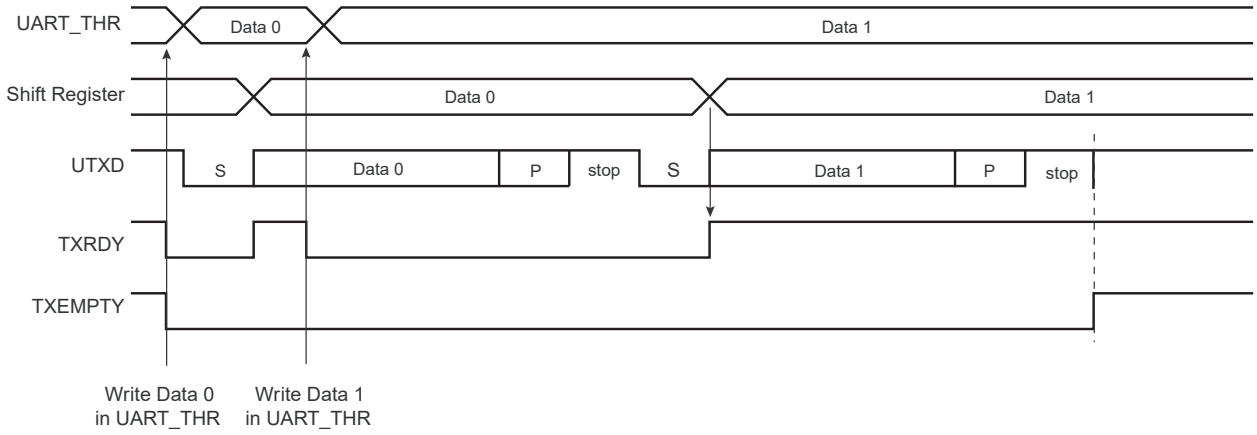



#### 47.5.3.3 Transmitter Control

When the transmitter is enabled, the bit TXRDY (Transmitter Ready) is set in UART_SR. The transmission starts when the programmer writes in the UART_THR, and after the written character is transferred from UART_THR to the internal shift register. The TXRDY bit remains high until a second character is written in UART_THR. As soon as the first character is completed, the last character written in UART_THR is transferred into the internal shift register and TXRDY rises again, showing that the holding register is empty.

When both the internal shift register and UART_THR are empty, i.e., all the characters written in UART_THR have been processed, the TXEMPTY bit rises after the last stop bit has been completed.

**Figure 47-11. Transmitter Control**




### 47.5.4 DMA Support

Both the receiver and the transmitter of the UART are connected to a DMA Controller (DMAC) channel.

The DMA Controller channels are programmed via registers that are mapped within the DMAC user interface.

### 47.5.5 Comparison Function on Received Character

When a comparison is performed on a received character, the result of the comparison is reported on the CMP flag in UART_SR when UART_RHR is loaded with the new received character. The CMP flag is cleared by writing a one to the RSTSTA bit in UART_CR.

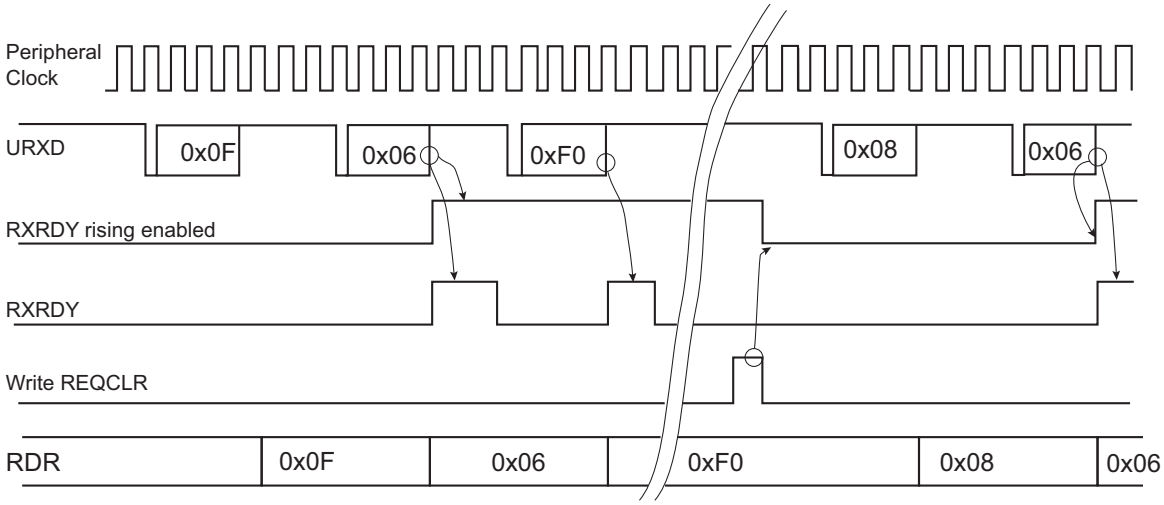
UART_CMPR (see UART Comparison Register) can be programmed to provide different comparison methods. These are listed below:

- If VAL1 equals VAL2, then the comparison is performed on a single value and the flag is set to 1 if the received character equals VAL1.
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 sets the CMP flag.
- If VAL1 is strictly higher than VAL2, then the flag CMP is set to 1 if either received character equals VAL1 or VAL2.

By programming the CMPMODE bit to 1, the comparison function result triggers the start of the loading of UART_RHR (see the figure below). The trigger condition occurs as soon as the received character value matches the condition defined by the programming of VAL1, VAL2 and CMPPAR in UART_CMPR. The comparison trigger event can be restarted by writing a one to the REQCLR bit in UART_CR.

**Figure 47-12. Receive Holding Register Management**

CMPMODE = 1, VAL1 = VAL2 = 0x06




### 47.5.6 Asynchronous and Partial Wake-up (SleepWalking)

Asynchronous and partial wake-up (SleepWalking) is a means of data pre-processing that qualifies an incoming event, thus allowing the UART to decide whether or not to wake up the system. SleepWalking is used primarily when the system is in Wait mode (refer to section “Power Management Controller (PMC)”) but can also be enabled when the system is fully running.

No access must be performed in the UART between the enable of asynchronous partial wake-up and the wake-up performed by the UART.

If the system is in Wait mode and asynchronous and partial wake-up is enabled, the maximum baud rate that can be achieved equals 19200.

If the system is running or in Sleep mode, the maximum baud rate that can be achieved equals 115200 or higher. This limit is bounded by the peripheral clock frequency divided by 16.

The UART_RHR must be read before enabling asynchronous and partial wake-up.

When SleepWalking is enabled for the UART (refer to section “Power Management Controller (PMC)”), the PMC decodes a clock request from the UART. The request is generated as soon as there is a falling edge on the URXD line as this may indicate the beginning of a start bit. If the system is in Wait mode (processor and peripheral clocks switched off), the PMC restarts the fast RC oscillator and provides the clock only to the UART.

As soon as the clock is provided by the PMC, the UART processes the received frame and compares the received character with VAL1 and VAL2 in UART_CMPR (UART Comparison Register).

The UART instructs the PMC to disable the clock if the received character value does not meet the conditions defined by VAL1 and VAL2 fields in UART_CMPR (see Asynchronous Event Generating Only Partial Wake-up).

If the received character value meets the conditions, the UART instructs the PMC to exit the full system from Wait mode (see Asynchronous Wake-up Use Case Examples).

The VAL1 and VAL2 fields can be programmed to provide different comparison methods and thus matching conditions.

- If VAL1 equals VAL2, then the comparison is performed on a single value and the wake-up is triggered if the received character equals VAL1.
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 wakes up the system.
- If VAL1 is strictly higher than VAL2, then the wake-up is triggered if the received character equals VAL1 or VAL2.
- If VAL1 = 0 and VAL2 = 255, the wake-up is triggered as soon as a character is received.

The matching condition can be configured to include the parity bit (CMPPAR in UART_CMPR). Thus, if the received data matches the comparison condition defined by VAL1 and VAL2 but a parity error is encountered, the matching condition is cancelled and the UART instructs the PMC to disable the clock (see Asynchronous Event Generating Only Partial Wake-up).

If the processor and peripherals are running, the UART can be configured in Asynchronous and partial wake-up mode by enabling the PMC_SLPWK_ER (refer to "Power Management Controller (PMC)"). When activity is detected on the receive line, the UART requests the clock from the PMC and the comparison is performed. If there is a comparison match, the UART continues to request the clock. If there is no match, the clock is switched off for the UART only, until a new activity is detected.

The CMPMODE configuration has no effect when Asynchronous and Partial Wake-up mode is enabled for the UART (refer to PMC_SLPWK_ER in "Power Management Controller (PMC)").

When the system is kept in active/running mode and the UART enters Asynchronous and Partial Wake-up mode, the flag CMP must be programmed as the unique source of the UART interrupt.

When the system exits Wait mode as the result of a matching condition, the RXRDY flag is used to determine if the UART is the source of exit.

**Note:** If the SleepWalking function is enabled on the UART, a divide by 8 of the peripheral clock versus the bus clock is not possible. Other dividers can be used with no constraints.

**Figure 47-13. Asynchronous Wake-up Use Case Examples**

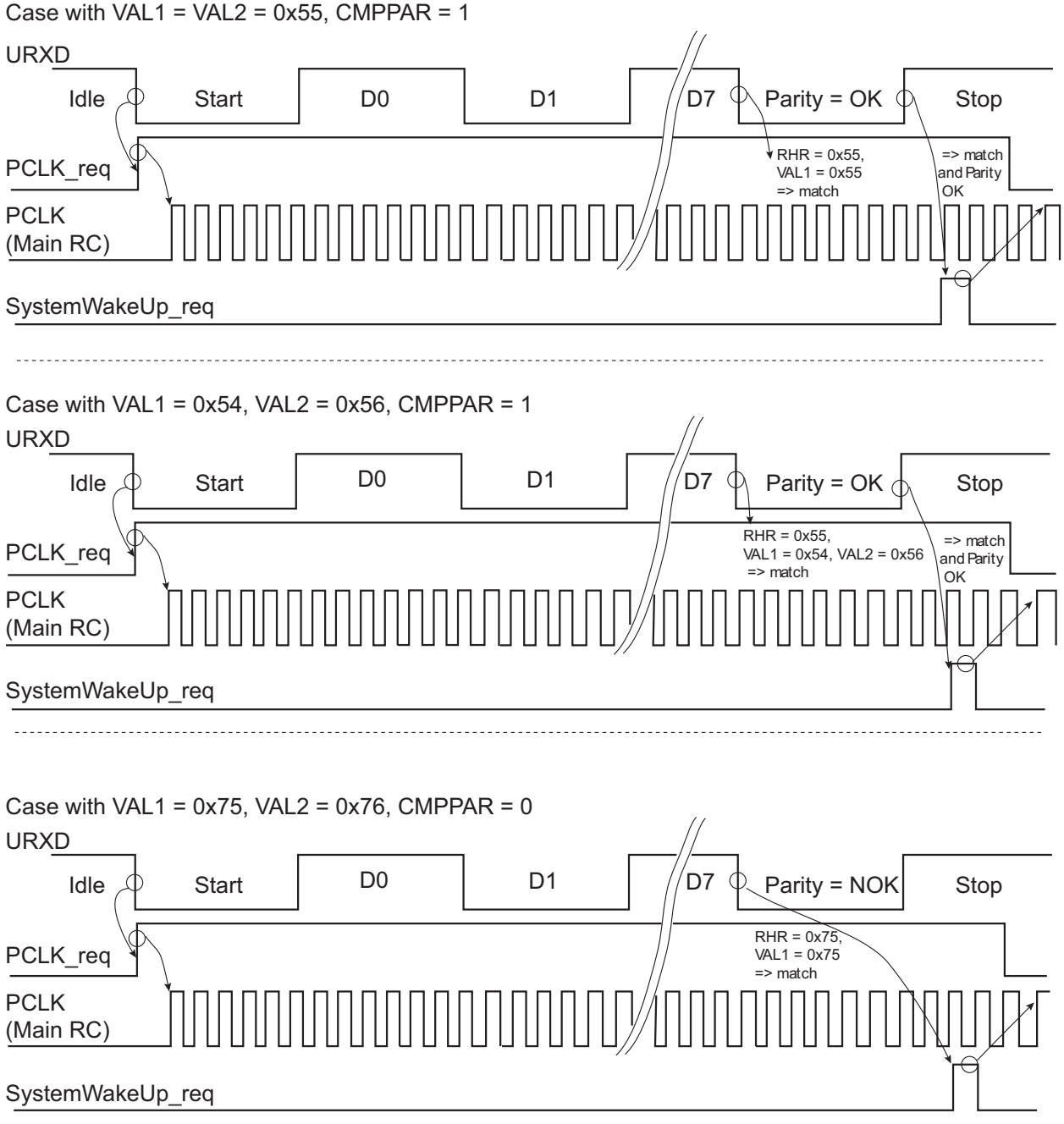
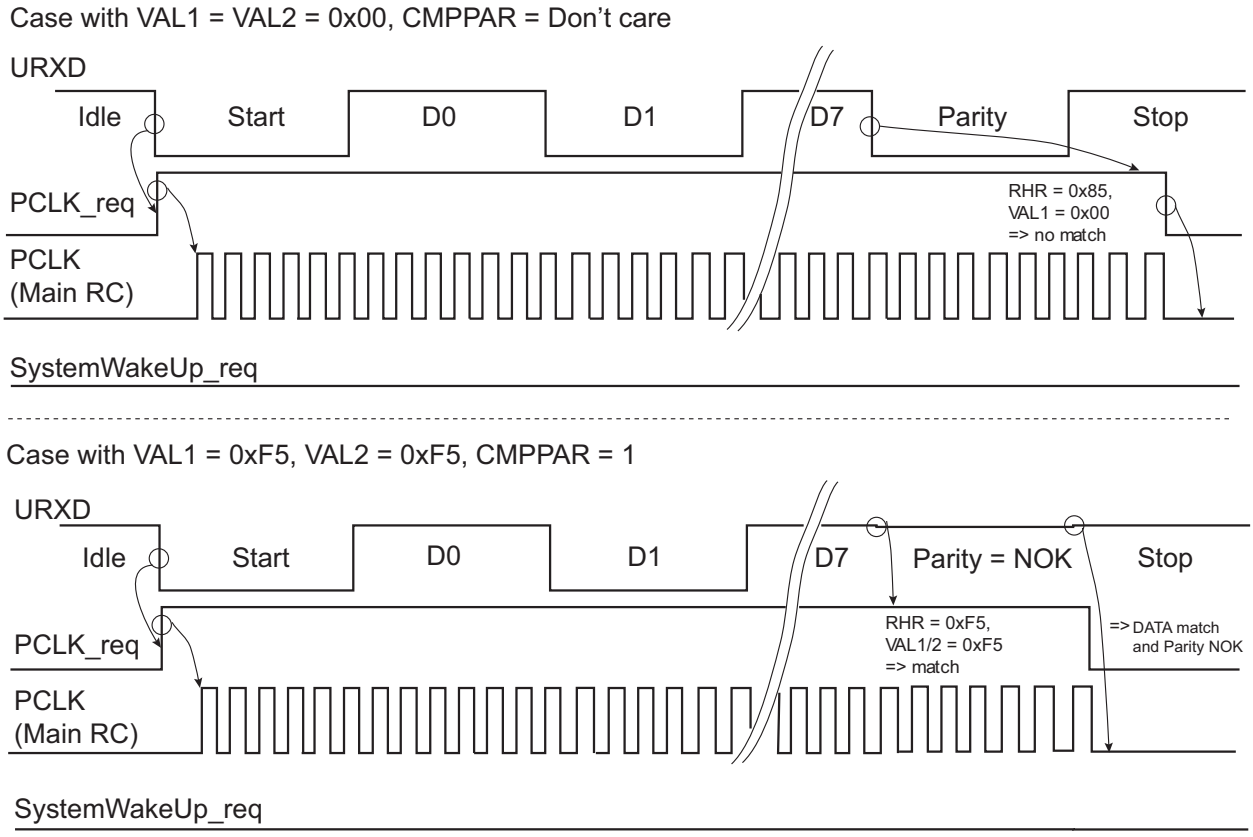


**Figure 47-14. Asynchronous Event Generating Only Partial Wake-up**




### 47.5.7 Register Write Protection

To prevent any single software error from corrupting UART behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the UART Write Protection Mode Register (UART_WPMR).

The following registers can be write-protected:

- UART Mode Register
- UART Baud Rate Generator Register
- UART Comparison Register
- UART Receiver Time-out Register

### 47.5.8 Test Modes

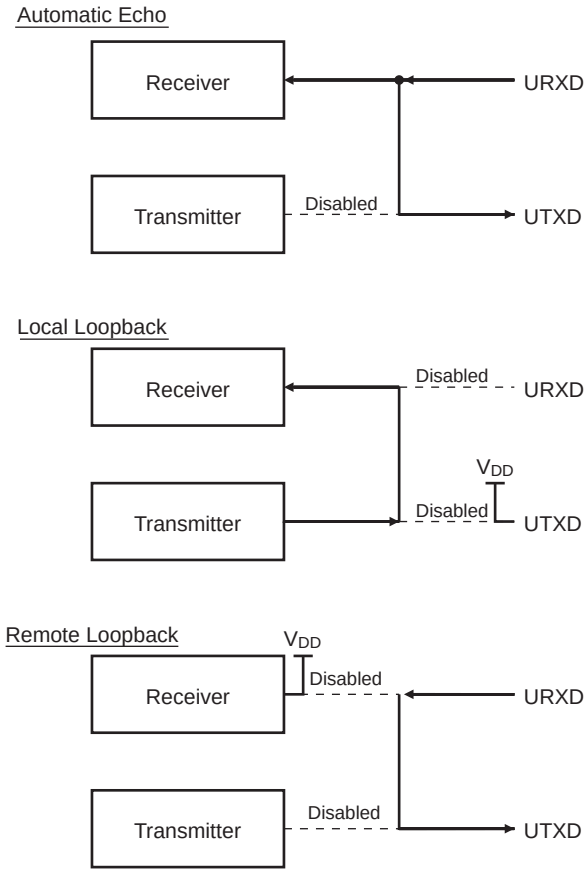
The UART supports three test modes. These modes of operation are programmed by using the CHMODE field in UART_MR.

The Automatic Echo mode allows a bit-by-bit retransmission. When a bit is received on the URXD line, it is sent to the UTXD line. The transmitter operates normally, but has no effect on the UTXD line.

The Local Loopback mode allows the transmitted characters to be received. UTXD and URXD pins are not used and the output of the transmitter is internally connected to the input of the receiver. The URXD pin level has no effect and the UTXD line is held high, as in idle state.

The Remote Loopback mode directly connects the URXD pin to the UTXD line. The transmitter and the receiver are disabled and have no effect. This mode allows a bit-by-bit retransmission.

**Figure 47-15. Test Modes**



## 47.6 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | UART_CR | 7:0 | TXDIS | TXEN | RXDIS | RXEN | RSTTX | RSTRX | | |
| | | 15:8 | | | | REQCLR | STTTO | RETTO | | RSTSTA |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | UART_MR | 7:0 | | | | FILTER | | | | |
| | | 15:8 | CHMODE[1:0] | | | BRSRCCK | PAR[2:0] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | UART_IER | 7:0 | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | CMP | | | | | | TXEMPTY | TIMEOUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | UART_IDR | 7:0 | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | CMP | | | | | | TXEMPTY | TIMEOUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | UART_IMR | 7:0 | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | CMP | | | | | | TXEMPTY | TIMEOUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | UART_SR | 7:0 | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | CMP | | | | | | TXEMPTY | TIMEOUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | UART_RHR | 7:0 | RXCHR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | UART_THR | 7:0 | TXCHR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | UART_BRGR | 7:0 | CD[7:0] | | | | | | | |
| | | 15:8 | CD[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 | UART_CMPR | 7:0 | VAL1[7:0] | | | | | | | |
| | | 15:8 | | CMPPAR | | CMPMODE | | | | |
| | | 23:16 | VAL2[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x28 | UART_RTOR | 7:0 | TO[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x2C ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | UART_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |

### 47.6.1 UART Control Register

**Name:** UART_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | REQCLR | STTTO | RETTO | | RSTSTA |
| Access | | | | W | W | W | | W |
| Reset | | | | – | – | – | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXDIS | TXEN | RXDIS | RXEN | RSTTX | RSTRX | | |
| Access | W | W | W | W | W | W | | |
| Reset | – | – | – | – | – | – | | |

**Bit 12 – REQCLR** Request Clear

- SleepWalking enabled:

0: No effect.

1: Bit REQCLR clears the potential clock request currently issued by UART, thus the potential system wake-up is cancelled.

- SleepWalking disabled:

0: No effect.

1: Bit REQCLR restarts the comparison trigger to enable receive holding register loading.

**Bit 11 – STTTO** Start Time-out

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Starts waiting for a character before clocking the time-out counter. Resets status bit TIMEOUT in UART_SR. |

**Bit 10 – RETTO** Rearm Time-out

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Restarts time-out. |

**Bit 8 – RSTSTA** Reset Status

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the status bits PARE, FRAME, CMP and OVRE in the UART_SR. |

**Bit 7 – TXDIS** Transmitter Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The transmitter is disabled. If a character is being processed and a character has been written in the UART_THR and RSTTX is not set, both characters are completed before the transmitter is stopped. |

**Bit 6 – TXEN** Transmitter Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The transmitter is enabled if TXDIS is 0. |

**Bit 5 – RXDIS** Receiver Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The receiver is disabled. If a character is being processed and RSTRX is not set, the character is completed before the receiver is stopped. |

**Bit 4 – RXEN** Receiver Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The receiver is enabled if RXDIS is 0. |

**Bit 3 – RSTTX** Reset Transmitter

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The transmitter logic is reset and disabled. If a character is being transmitted, the transmission is aborted. |

**Bit 2 – RSTRX** Reset Receiver

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The receiver logic is reset and disabled. If a character is being received, the reception is aborted. |

### 47.6.2 UART Mode Register

**Name:** UART_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CHMODE[1:0] | | | BRSRCCK | PAR[2:0] | | | |
| Access | R/W | R/W | | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | | 0 | 0 | 0 | 0 | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | FILTER | | | | |
| Access | | | | R/W | | | | |
| Reset | | | | 0 | | | | |

**Bits 15:14 – CHMODE[1:0]** Channel Mode

| Value | Name | Description |
|---|---|---|
| 0 | NORMAL | Normal mode |
| 1 | AUTOMATIC | Automatic echo |
| 2 | LOCAL_LOOPBACK | Local loopback |
| 3 | REMOTE_LOOPBACK | Remote loopback |

**Bit 12 – BRSRCCK** Baud Rate Source Clock
0 (PERIPH_CLK): The baud rate is driven by the peripheral clock

1 (GCLK): The baud rate is driven by a PMC-programmable clock GCLK (refer to section "Power Management Controller (PMC)").

**Bits 11:9 – PAR[2:0]** Parity Type

| Value | Name | Description |
|---|---|---|
| 0 | EVEN | Even Parity |
| 1 | ODD | Odd Parity |
| 2 | SPACE | Space: parity forced to 0 |
| 3 | MARK | Mark: parity forced to 1 |
| 4 | NO | No parity |

**Bit 4 – FILTER** Receiver Digital Filter
0 (DISABLED): UART does not filter the receive line.

1 (ENABLED): UART filters the receive line using a three-sample filter (16x-bit clock) (2 over 3 majority).

### 47.6.3 UART Interrupt Enable Register

**Name:** UART_IER
**Offset:** 0x08
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMP | | | | | | TXEMPTY | TIMEOUT |
| Access | W | | | | | | W | W |
| Reset | – | | | | | | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| Access | W | W | W | | | | W | W |
| Reset | – | – | – | | | | – | – |

**Bit 15 – CMP** Enable Comparison Interrupt

**Bit 9 – TXEMPTY** Enable TXEMPTY Interrupt

**Bit 8 – TIMEOUT** Enable Time-out Interrupt

**Bit 7 – PARE** Enable Parity Error Interrupt

**Bit 6 – FRAME** Enable Framing Error Interrupt

**Bit 5 – OVRE** Enable Overrun Error Interrupt

**Bit 1 – TXRDY** Enable TXRDY Interrupt

**Bit 0 – RXRDY** Enable RXRDY Interrupt

### 47.6.4 UART Interrupt Disable Register

**Name:** UART_IDR
**Offset:** 0x0C
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMP | | | | | | TXEMPTY | TIMEOUT |
| Access | W | | | | | | W | W |
| Reset | – | | | | | | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| Access | W | W | W | | | | W | W |
| Reset | – | – | – | | | | – | – |

**Bit 15 – CMP** Disable Comparison Interrupt

**Bit 9 – TXEMPTY** Disable TXEMPTY Interrupt

**Bit 8 – TIMEOUT** Disable Time-out Interrupt

**Bit 7 – PARE** Disable Parity Error Interrupt

**Bit 6 – FRAME** Disable Framing Error Interrupt

**Bit 5 – OVRE** Disable Overrun Error Interrupt

**Bit 1 – TXRDY** Disable TXRDY Interrupt

**Bit 0 – RXRDY** Disable RXRDY Interrupt

### 47.6.5 UART Interrupt Mask Register

**Name:** UART_IMR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMP | | | | | | TXEMPTY | TIMEOUT |
| Access | R | | | | | | R | R |
| Reset | 0 | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| Access | R | R | R | | | | R | R |
| Reset | 0 | 0 | 0 | | | | 0 | 0 |

**Bit 15 – CMP** Mask Comparison Interrupt

**Bit 9 – TXEMPTY** Mask TXEMPTY Interrupt

**Bit 8 – TIMEOUT** Mask Time-out Interrupt

**Bit 7 – PARE** Mask Parity Error Interrupt

**Bit 6 – FRAME** Mask Framing Error Interrupt

**Bit 5 – OVRE** Mask Overrun Error Interrupt

**Bit 1 – TXRDY** Disable TXRDY Interrupt

**Bit 0 – RXRDY** Mask RXRDY Interrupt

### 47.6.6 UART Status Register

**Name:** UART_SR
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMP | | | | | | TXEMPTY | TIMEOUT |
| Access | R | | | | | | R | R |
| Reset | 0 | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| Access | R | R | R | | | | R | R |
| Reset | 0 | 0 | 0 | | | | 0 | 0 |

**Bit 15 – CMP** Comparison Match

| Value | Description |
|---|---|
| 0 | No received character matches the comparison criteria programmed in VAL1, VAL2 fields and in CMPPAR bit since the last RSTSTA. |
| 1 | The received character matches the comparison criteria. |

**Bit 9 – TXEMPTY** Transmitter Empty

| Value | Description |
|---|---|
| 0 | There are characters in UART_THR, or characters being processed by the transmitter, or the transmitter is disabled. |
| 1 | There are no characters in UART_THR and there are no characters being processed by the transmitter. |

**Bit 8 – TIMEOUT** Receiver Time-out

| Value | Description |
|---|---|
| 0 | There has not been a time-out since the last Start Time-out command (STTTO in UART_CR) or the Time-out Register is 0. |
| 1 | There has been a time-out since the last Start Time-out command (STTTO in UART_CR). |

**Bit 7 – PARE** Parity Error

| Value | Description |
|---|---|
| 0 | No parity error has occurred since the last RSTSTA. |
| 1 | At least one parity error has occurred since the last RSTSTA. |

**Bit 6 – FRAME** Framing Error

| Value | Description |
|---|---|
| 0 | No framing error has occurred since the last RSTSTA. |
| 1 | At least one framing error has occurred since the last RSTSTA. |

**Bit 5 – OVRE** Overrun Error

| Value | Description |
|---|---|
| 0 | No overrun error has occurred since the last RSTSTA. |
| 1 | At least one overrun error has occurred since the last RSTSTA. |

**Bit 1 – TXRDY** Transmitter Ready

| Value | Description |
|---|---|
| 0 | A character has been written to UART_THR and not yet transferred to the internal shift register, or the transmitter is disabled. |
| 1 | There is no character written to UART_THR not yet transferred to the internal shift register. |

**Bit 0 – RXRDY** Receiver Ready

| Value | Description |
|---|---|
| 0 | No character has been received since the last read of the UART_RHR, or the receiver is disabled. |
| 1 | At least one complete character has been received, transferred to UART_RHR and not yet read. |

### 47.6.7 UART Receiver Holding Register

**Name:** UART_RHR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXCHR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – RXCHR[7:0]** Received Character
Last received character if RXRDY is set.

### 47.6.8 UART Transmit Holding Register

**Name:** UART_THR
**Offset:** 0x1C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXCHR[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 7:0 – TXCHR[7:0]** Character to be Transmitted
Next character to be transmitted after the current character if TXRDY is not set.

### 47.6.9 UART Baud Rate Generator Register

**Name:** UART_BRGR
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CD[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – CD[15:0]** Clock Divisor

| Value | Description |
|---|---|
| 0 | Baud rate clock is disabled |
| 1 to 65,535 | If BRSRCCK = 0:<br>$\mathrm{CD} = \frac{\mathrm{f}_{\text{peripheral clock}}}{16 \times \text{Baud Rate}}$<br>If BRSRCCK = 1:<br>$\mathrm{CD} = \frac{\mathrm{f}_{\mathrm{GCLKx}}}{16 \times \text{Baud Rate}}$ |

### 47.6.10 UART Comparison Register

**Name:** UART_CMPR
**Offset:** 0x24
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | VAL2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | CMPPAR | | CMPMODE | | | | |
| Access | | R/W | | R/W | | | | |
| Reset | | 0 | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | VAL1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – VAL2[7:0]** Second Comparison Value for Received Character

| Value | Description |
|---|---|
| 0-255 | The received character must be lower or equal to the value of VAL2 and higher or equal to VAL1 to set CMP flag in UART_SR. If asynchronous partial wake-up (SleepWalking) is enabled in PMC_SLPWK_ER, the UART requests a system wake-up if condition is met. |

**Bit 14 – CMPPAR** Compare Parity

| Value | Description |
|---|---|
| 0 | The parity is not checked and a bad parity cannot prevent from waking up the system. |
| 1 | The parity is checked and a matching condition on data can be cancelled by an error on parity bit, so no wake-up is performed. |

**Bit 12 – CMPMODE** Comparison Mode

| Value | Name | Description |
|---|---|---|
| 0 | FLAG_ONLY | Any character is received and comparison function drives CMP flag. |
| 1 | START_CONDITION | Comparison condition must be met to start reception. |

**Bits 7:0 – VAL1[7:0]** First Comparison Value for Received Character

| Value | Description |
|---|---|
| 0–255 | The received character must be higher or equal to the value of VAL1 and lower or equal to VAL2 to set CMP flag in UART_SR. If asynchronous partial wake-up (SleepWalking) is enabled in PMC_SLPWK_ER, the UART requests a system wake-up if the condition is met. |

### 47.6.11 UART Receiver Time-out Register

**Name:** UART_RTOR
**Offset:** 0x28
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TO[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – TO[7:0]** Time-out Value

| Value | Description |
|---|---|
| 0 | The receiver time-out is disabled. |
| 1–255 | The receiver time-out is enabled and the time-out delay is TO x bit period. |

### 47.6.12 UART Write Protection Mode Register

**Name:** UART_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x554152 | PASSWD | Writing any other value in this field aborts the write operation. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See Register Write Protection for the list of registers that can be protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x554152 (UART in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x554152 (UART in ASCII). |

# 48. Serial Peripheral Interface (SPI)

## 48.1 Description

The Serial Peripheral Interface (SPI) circuit is a synchronous serial data link that provides communication with external devices in Master or Slave mode. It also enables communication between processors if an external processor is connected to the system.

The Serial Peripheral Interface is essentially a shift register that serially transmits data bits to other SPIs. During a data transfer, one SPI system acts as the “master”' which controls the data flow, while the other devices act as “slaves" which have data shifted into and out by the master. Different CPUs can take turn being masters (multiple master protocol, contrary to single master protocol where one CPU is always the master while all of the others are always slaves). One master can simultaneously shift data into multiple slaves. However, only one slave can drive its output to write data back to the master at any given time.

A slave device is selected when the master asserts its NSS signal. If multiple slave devices exist, the master generates a separate slave select signal for each slave (NPCS).

The SPI system consists of two data lines and two control lines:

- Master Out Slave In (MOSI)—This data line supplies the output data from the master shifted into the input(s) of the slave(s).
- Master In Slave Out (MISO)—This data line supplies the output data from a slave to the input of the master. There may be no more than one slave transmitting data during any particular transfer.
- Serial Clock (SPCK)—This control line is driven by the master and regulates the flow of the data bits. The master can transmit data at a variety of baud rates; there is one SPCK pulse for each bit that is transmitted.
- Slave Select (NSS)—This control line allows slaves to be turned on and off by hardware.

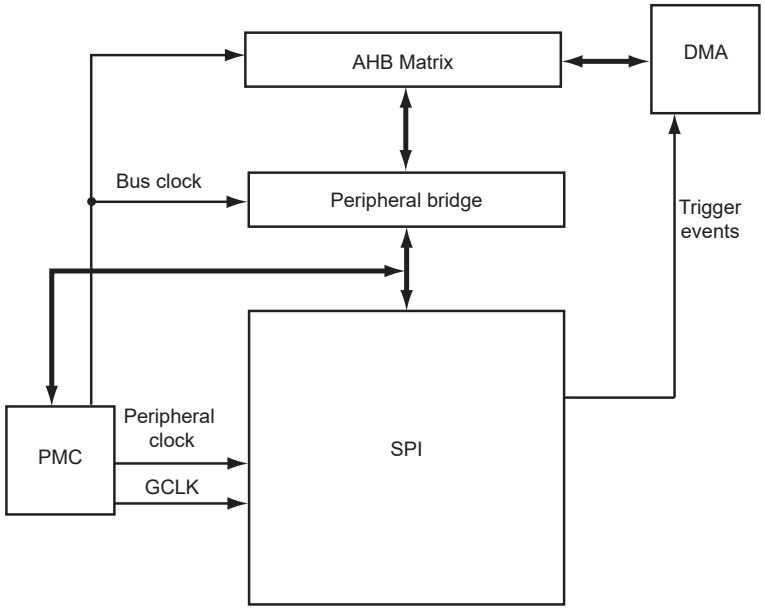
## 48.2 Embedded Characteristics

- Master or Slave Serial Peripheral Bus Interface
  - 8-bit to 16-bit programmable data length per chip select
  - Programmable phase and polarity per chip select
  - Programmable transfer delay between consecutive transfers and delay before SPI clock per chip select
  - Programmable delay between chip selects
  - Selectable mode fault detection
- Master Mode can Drive SPCK up to Peripheral Clock
- 16-data Transmit and Receive FIFOs
- Master Mode Bit Rate can be Independent of the Processor/Peripheral Clock
- Slave Mode Operates on SPCK, Asynchronously with Core and Bus Clock
- Four Chip Selects with External Decoder Support Allow Communication with up to 15 Peripherals
- Communication with Serial External Devices Supported
  - Serial memories, such as DataFlash and 3-wire EEPROMs
  - Serial peripherals, such as ADCs, DACs, LCD controllers, CAN controllers and sensors
  - External coprocessors

- Connection to DMA Channel Capabilities, Optimizing Data Transfers
  - One channel for the receiver
  - One channel for the transmitter
- Register Write Protection

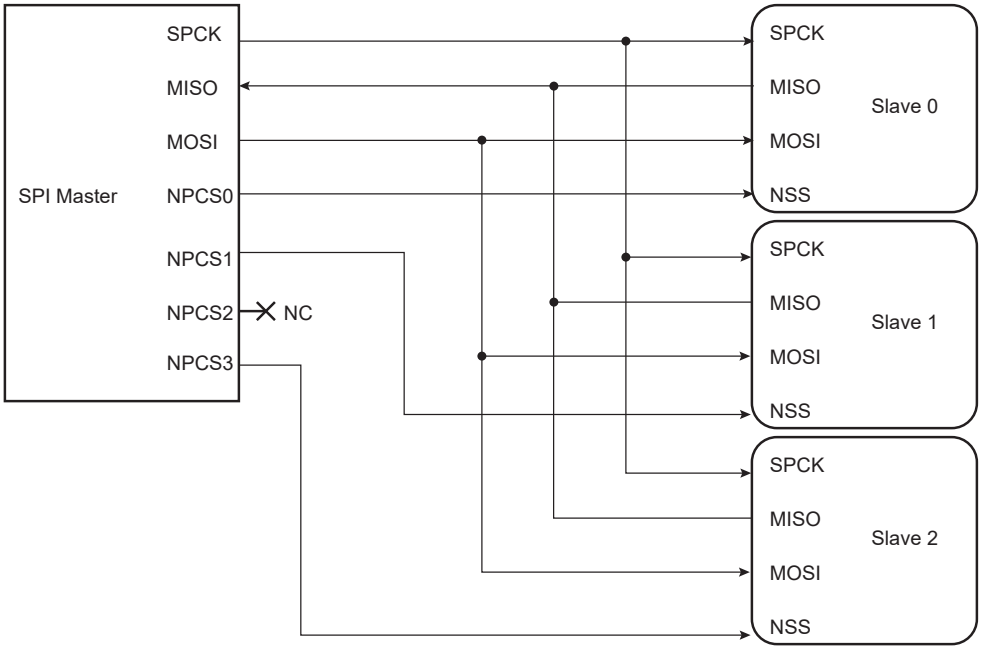
## 48.3 Block Diagram

**Figure 48-1. Block Diagram**




## 48.4 Application Block Diagram

**Figure 48-2. Application Block Diagram: Single Master/Multiple Slave Implementation**



## 48.5 Signal Description

**Table 48-1. Signal Description**

| Pin Name | Pin Description | Type | |
|---|---|---|---|
| | | Master | Slave |
| MISO | Master In Slave Out | Input | Output |
| MOSI | Master Out Slave In | Output | Input |
| SPCK | Serial Clock | Output | Input |
| NPCS1–NPCS3 | Peripheral Chip Selects | Output | Unused |
| NPCS0/NSS | Peripheral Chip Select/Slave Select | Output | Input |

## 48.6 Product Dependencies

### 48.6.1 I/O Lines

The pins used for interfacing the compliant external devices can be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the SPI pins to their peripheral functions.

### 48.6.2 Power Management

The SPI can be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the SPI clock.

### 48.6.3 Interrupt

The SPI interface has an interrupt line connected to the interrupt controller. Handling the SPI interrupt requires programming the interrupt controller before configuring the SPI.

### 48.6.4 Direct Memory Access Controller (DMAC)

The SPI interface can be used in conjunction with the DMAC in order to reduce processor overhead. For a full description of the DMAC, refer to the relevant section.

## 48.7 Functional Description

### 48.7.1 Modes of Operation

The SPI operates in Master mode or in Slave mode.

- The SPI operates in Master mode by setting the MSTR bit in the SPI Mode Register (SPI_MR):
  - Pins NPCS0 to NPCS3 are all configured as outputs
  - The SPCK pin is driven
  - The MISO line is wired on the receiver input
  - The MOSI line is driven as an output by the transmitter.
- The SPI operates in Slave mode if the MSTR bit in SPI_MR is written to ‘0’:
  - The MISO line is driven by the transmitter output
  - The MOSI line is wired on the receiver input

– The SPCK pin is driven by the transmitter to synchronize the receiver.
– The NPCS0 pin becomes an input, and is used as a slave select signal (NSS)
– The NPCS1 to NPCS3 pins are not driven and can be used for other purposes.

The data transfers are identically programmable for both modes of operation. The baud rate generator is activated only in Master mode.

### 48.7.2 Data Transfer

Four combinations of polarity and phase are available for data transfers. The clock polarity is programmed with the CPOL bit in the SPI Chip Select registers (SPI_CSRx). The clock phase is programmed with the NCPHA bit. These two parameters determine the edges of the clock signal on which data is driven and sampled. Each of the two parameters has two possible states, resulting in four possible combinations that are incompatible with one another. Consequently, a master/slave pair must use the same parameter pair values to communicate. If multiple slaves are connected and require different configurations, the master must reconfigure itself each time it needs to communicate with a different slave.

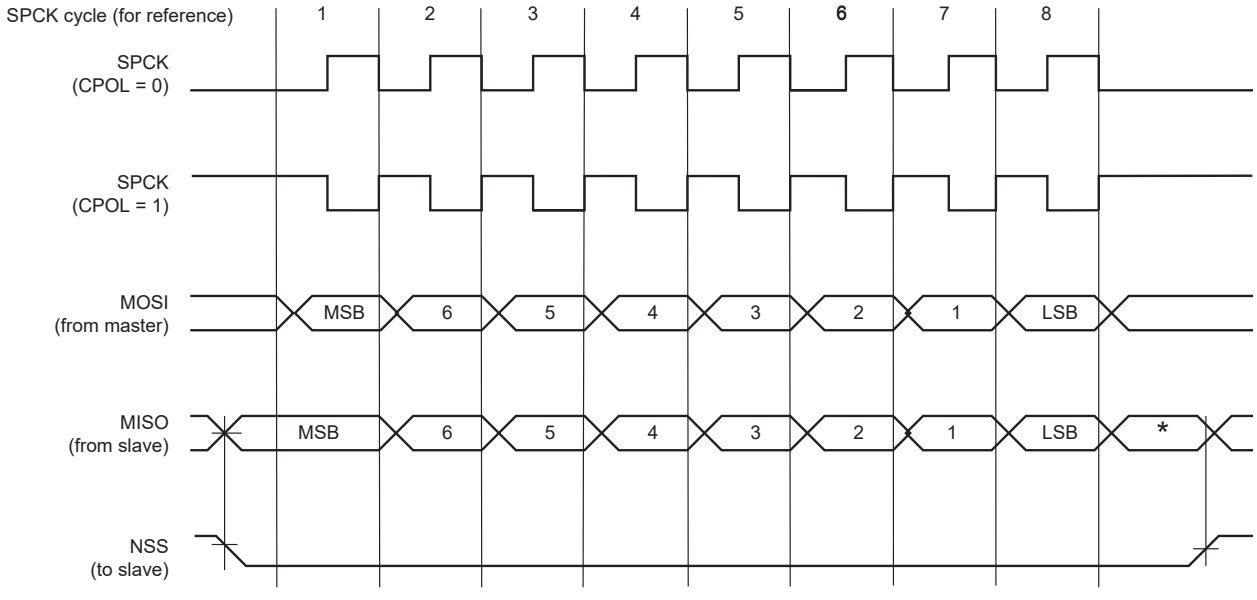
The table below shows the four modes and corresponding parameter settings.

**Table 48-2. SPI Bus Protocol Modes**

| SPI Mode | CPOL | NCPHA | Shift SPCK Edge | Capture SPCK Edge | SPCK Inactive Level |
|---|---|---|---|---|---|
| 0 | 0 | 1 | Falling | Rising | Low |
| 1 | 0 | 0 | Rising | Falling | Low |
| 2 | 1 | 1 | Rising | Falling | High |
| 3 | 1 | 0 | Falling | Rising | High |

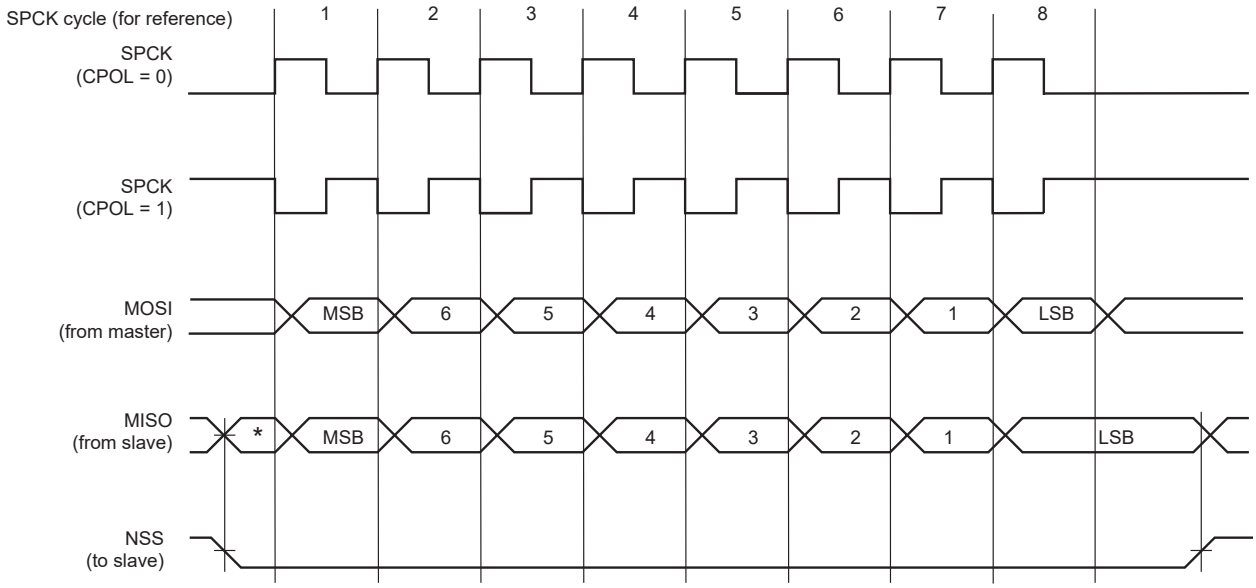
The following figures show examples of data transfers.

**Figure 48-3. SPI Transfer Format (NCPHA = 1, 8 bits per transfer)**




* Not defined.

**Figure 48-4. SPI Transfer Format (NCPHA = 0, 8 bits per transfer)**




* Not defined.

### 48.7.3 Master Mode Operations

When configured in Master mode, the SPI operates on the clock generated by the internal programmable baud rate generator. It fully controls the data transfers to and from the slave(s) connected to the SPI bus. The SPI drives the chip select line to the slave and the serial clock signal (SPCK).

The SPI features two holding registers, the Transmit Data Register (SPI_TDR) and the Receive Data Register (SPI_RDR), and a single shift register. The holding registers maintain the data flow at a constant rate.

After enabling the SPI, a data transfer starts when the processor writes to SPI_TDR. The written data is immediately transferred into the internal shift register and the transfer on the SPI bus starts. While the data in the shift register is shifted on the MOSI line, the MISO line is sampled and shifted into the shift register. Data cannot be loaded in SPI_RDR without transmitting data. If there is no data to transmit, dummy data can be used (SPI_TDR filled with ones). If SPI_MR.WDRBT is set, transmission can occur only if SPI_RDR has been read. If Receiving mode is not required, for example when communicating with a slave receiver only (such as an LCD), the receive status flags in the SPI Status register (SPI_SR) can be discarded.

Before writing SPI_TDR, SPI_MR.PCS must be set in order to select a slave.

If new data is written in SPI_TDR during the transfer, it is kept in SPI_TDR until the current transfer is completed. Then, the received data is transferred from the shift register to SPI_RDR, the data in SPI_TDR is loaded in the shift register and a new transfer starts.

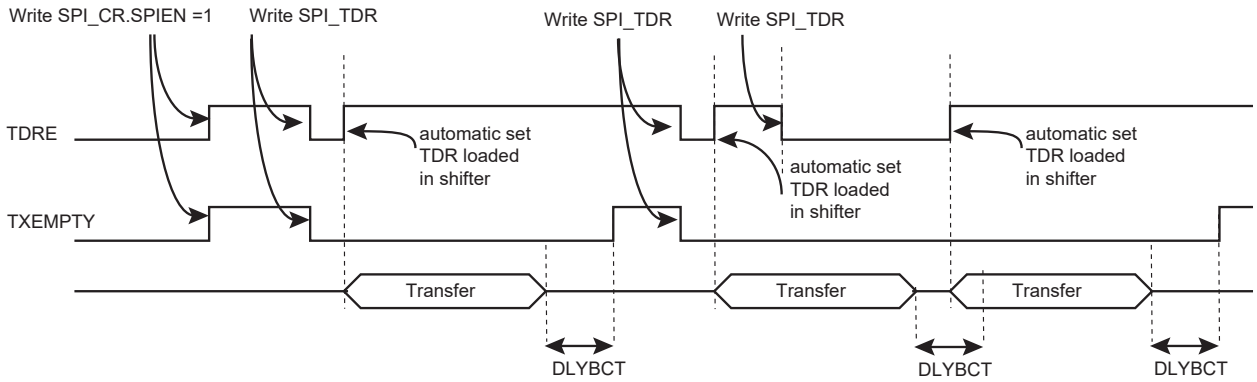
As soon as SPI_TDR is written, the Transmit Data Register Empty (TDRE) flag in SPI_SR is cleared. When the data written in SPI_TDR is loaded into the shift register, TDRE in SPI_SR is set. The TDRE flag is used to trigger the Transmit DMA channel.

See the figure below.

The end of transfer is indicated by the TXEMPTY flag in SPI_SR. If a transfer delay (DLYBCT) is greater than 0 for the last transfer, TXEMPTY is set after the completion of this delay. The peripheral clock can be switched off at this time.

**Note:** When the SPI is enabled, the TDRE and TXEMPTY flags are set.

**Figure 48-5. TDRE and TXEMPTY Flag Behavior**



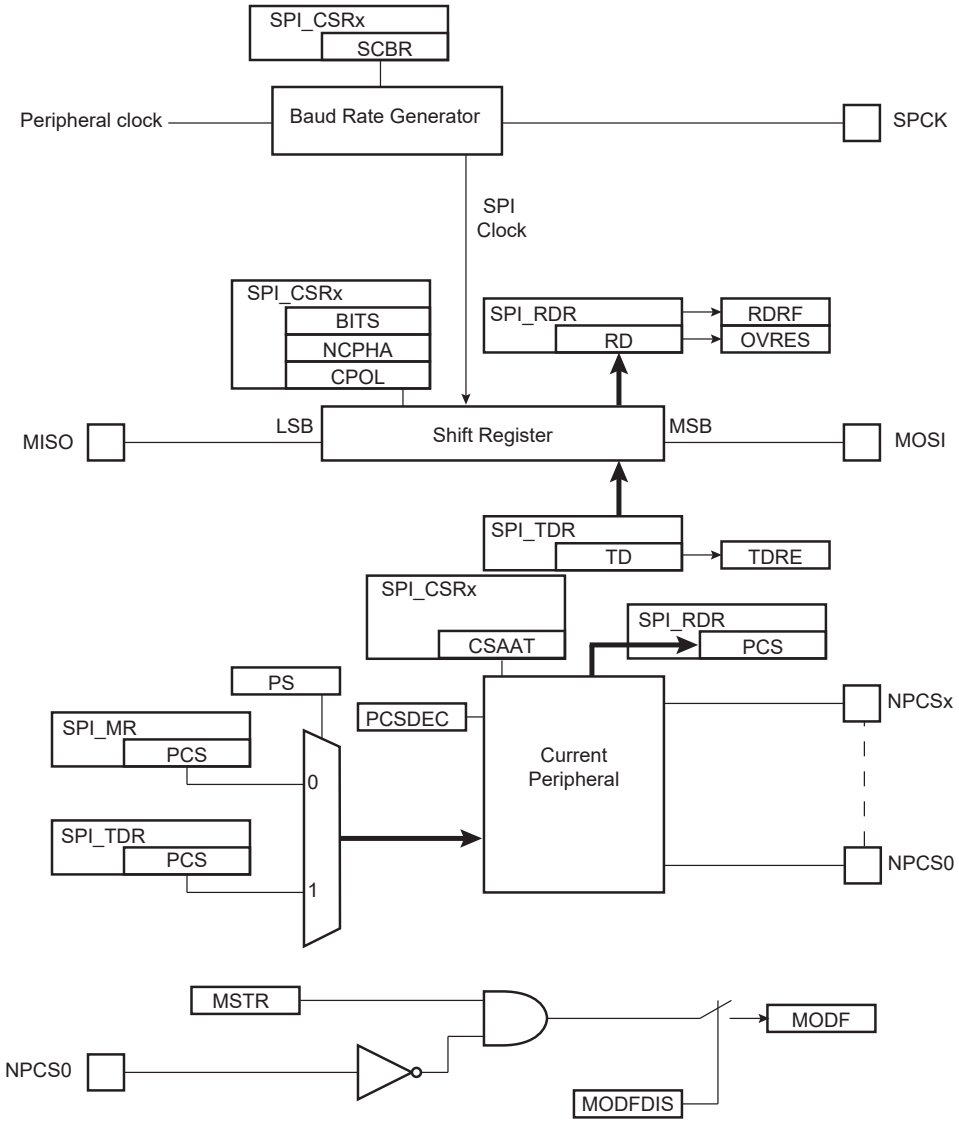

The transfer of received data from the internal shift register to SPI_RDR is indicated by the Receive Data Register Full (RDRF) bit in SPI_SR. When the received data is read, SPI_SR.RDRF is cleared.

If SPI_RDR has not been read before new data is received, the Overrun Error (OVRES) flag in SPI_SR is set. As long as this flag is set, data is loaded in SPI_RDR. The user has to read SPI_SR to clear OVRES.

The following figures show, respectively, a block diagram of the SPI when operating in Master mode and a flow chart describing how transfers are handled.

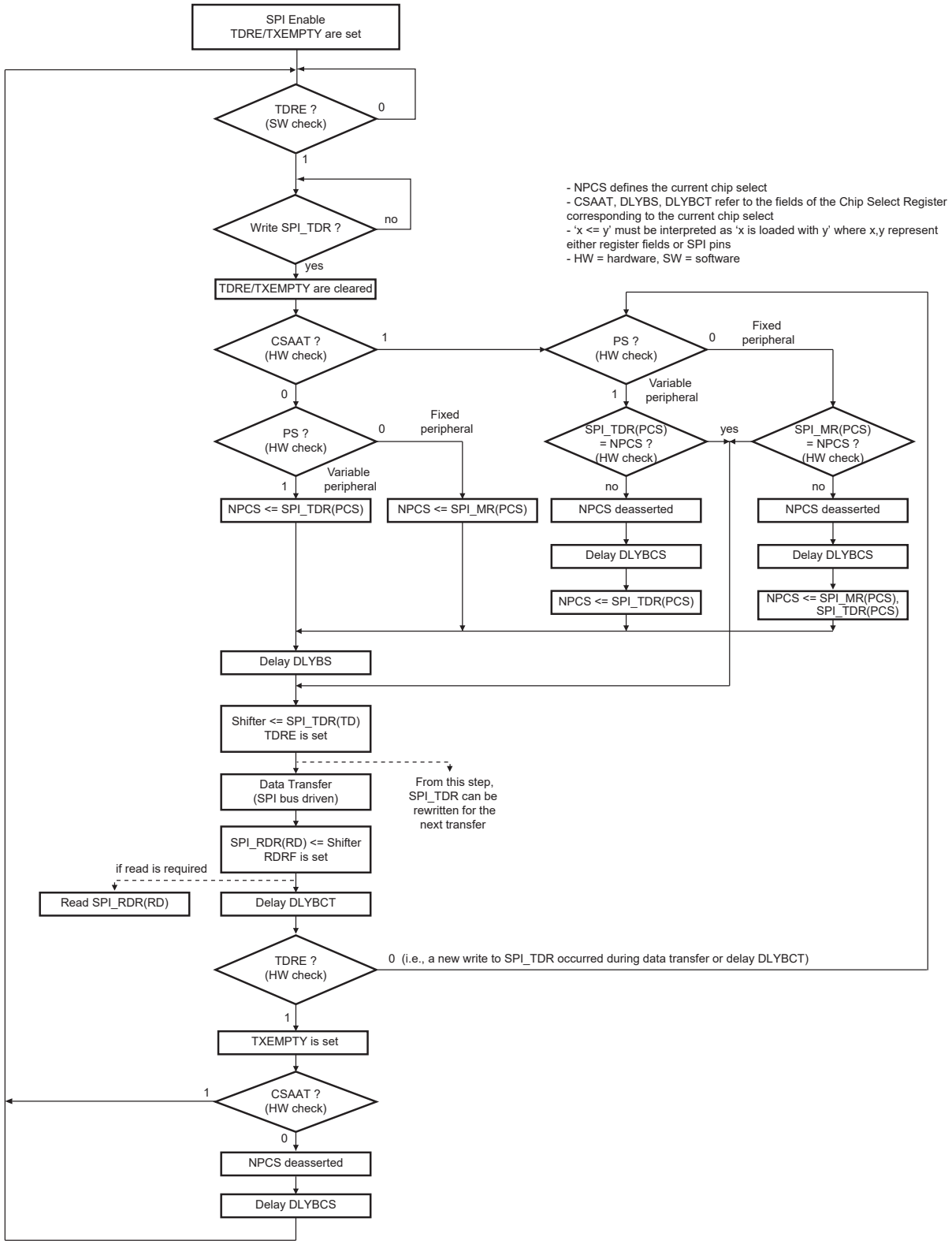
### 48.7.3.1 Master Mode Block Diagram

**Figure 48-6. Master Mode Block Diagram**



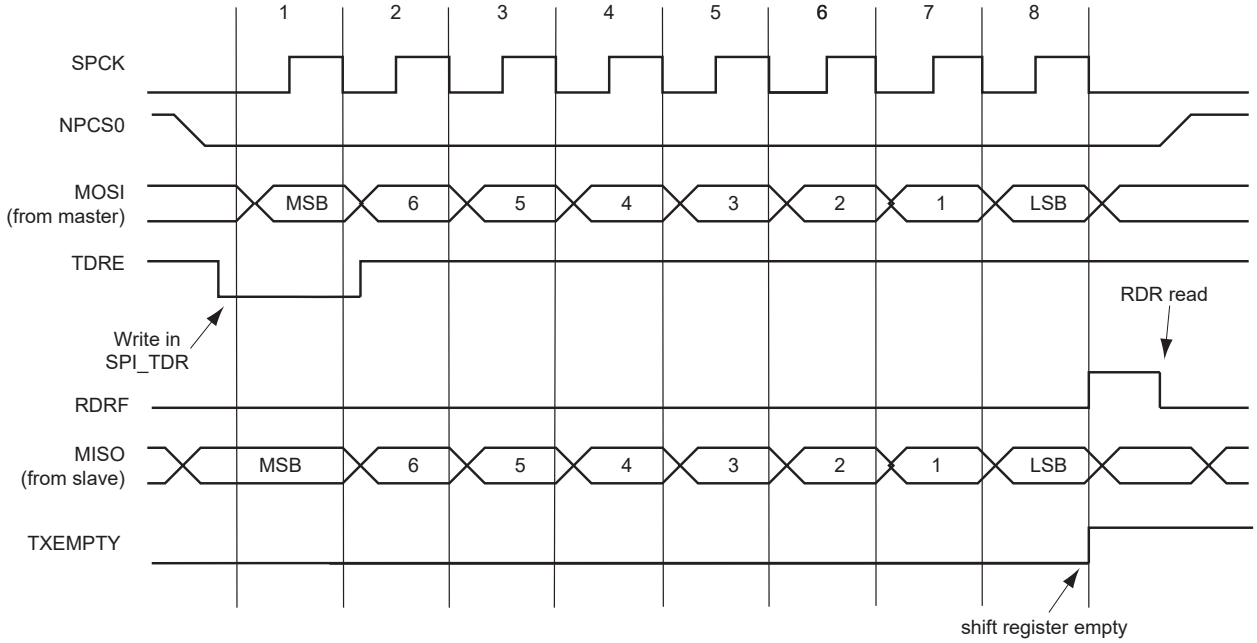
### 48.7.3.2 Master Mode Flow Diagram

**Figure 48-7. Master Mode Flow Diagram**



The figure below shows the behavior of Transmit Data Register Empty (TDRE), Receive Data Register (RDRF) and Transmission Register Empty (TXEMPTY) status flags within SPI_SR during an 8-bit data transfer in Fixed mode without the DMA involved.

**Figure 48-8. Status Register Flags Behavior**




### 48.7.3.3 Clock Generation

The SPI Baud rate clock is generated by dividing the peripheral clock by a value between 1 and 255.

If SPI_CSRx.SCBR is programmed to 1, the operating baud rate is peripheral clock (refer to the section “Electrical Characteristics” for the SPCK maximum frequency). Triggering a transfer while SPI_CSRx.SCBR is at 0 can lead to unpredictable results.

At reset, SPI_CSRx.SCBR=0 and the user has to program it to a valid value before performing the first transfer.

The divisor can be defined independently for each chip select, as it has to be programmed in SPI_CSRx.SCBR. This allows the SPI to automatically adapt the baud rate for each interfaced peripheral without reprogramming.

### 48.7.3.4 Transfer Delays

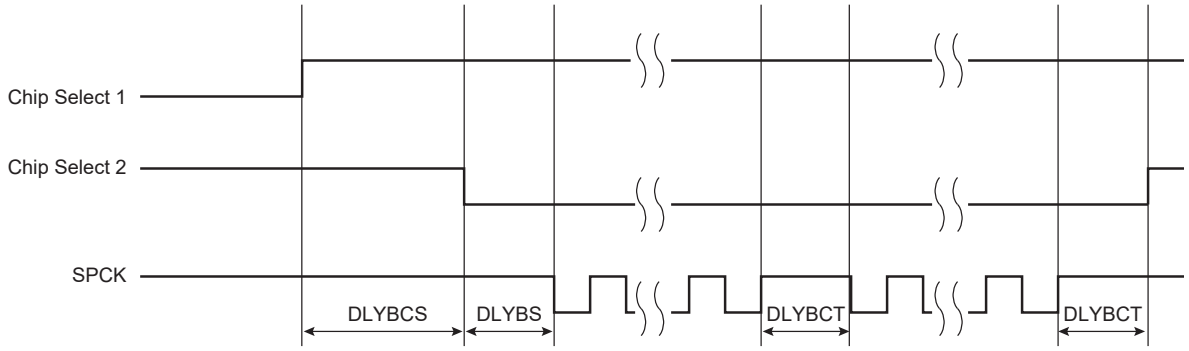
The following figure shows a chip select transfer change and consecutive transfers on the same chip select. Three delays can be programmed to modify the transfer waveforms:

- Delay between the chip selects—programmable only once for all chip selects by writing field SPI_MR.DLYBCS. The SPI slave device deactivation delay is managed through DLYBCS. If there is only one SPI slave device connected to the master, DLYBCS does not need to be configured. If several slave devices are connected to a master, DLYBCS must be configured depending on the highest deactivation delay. Refer to details on the SPI slave device in the section “Electrical Characteristics”.
- Delay before SPCK—independently programmable for each chip select by writing SPI_CSRx.DLYBS. The SPI slave device activation delay is managed through DLYBS. Refer to details on the SPI slave device in the section “Electrical Characteristics” to define DLYBS.

- Delay between consecutive transfers—independently programmable for each chip select by writing SPI_CSRx.DLYBCT. The time required by the SPI slave device to process received data is managed through DLYBCT. This time depends on the SPI slave system activity.

These delays allow the SPI to be adapted to the interfaced peripherals and their speed and bus release time.

**Figure 48-9. Programmable Delays**




### 48.7.3.5 Peripheral Selection

The serial peripherals are selected through the assertion of the NPCS0 to NPCS3 signals. By default, all NPCS signals are high before and after each transfer.

- Fixed Peripheral Select Mode: SPI exchanges data with only one peripheral.
  Fixed Peripheral Select mode is enabled by clearing SPI_MR.PS. In this case, the current peripheral is defined by SPI_MR.PCS. SPI_TDR.PCS has no effect.
- Variable Peripheral Select Mode: Data can be exchanged with more than one peripheral without having to reprogram SPI_MR.PCS.
  Variable Peripheral Select mode is enabled by setting SPI_MR.PS. SPI_TDR.PCS is used to select the current peripheral. This means that the peripheral selection can be defined for each new data. The value must be written in a single access to SPI_TDR in the following format:
  [xxxxxxx(7-bit) + LASTXFER(1-bit)(1)+ xxxx(4-bit) + PCS (4-bit) + TD (8- to 16-bit data)]

  with LASTXFER at 0 or 1 depending on the CSAAT bit, and PCS equal to the chip select to assert, as defined in section SPI Transmit Data Register.

  **Note:**
  1. Optional

  For details on CSAAT, LASTXFER and CSNAAT, see section Peripheral Deselection with another DMA.

  If LASTXFER is used, the command must be issued after writing the last character. Instead of LASTXFER, the user can use the SPIDIS command. After the end of the DMA transfer, it is necessary to wait for the TXEMPTY flag and then write SPIDIS into the SPI Control Register (SPI_CR). This does not change the configuration register values). The NPCS is disabled after the last character transfer. Then, another DMA transfer can be started if SPI_CR.SPIEN has previously been written.

### 48.7.3.6 SPI Direct Access Memory Controller (DMAC)

In both Fixed and Variable modes, the Direct Memory Access Controller (DMAC) can be used to reduce processor overhead.

The fixed peripheral selection allows buffer transfers with a single peripheral. Using the DMAC is an optimal means, as the size of the data transfer between the memory and the SPI is either 8 bits or 16 bits. However, if the peripheral selection is modified, SPI_MR must be reprogrammed.

The variable peripheral selection allows buffer transfers with multiple peripherals without reprogramming SPI_MR. Data written in SPI_TDR is 32 bits wide and defines the real data to be transmitted and the destination peripheral. Using the DMAC in this mode requires 32-bit wide buffers, with the data in the LSBs and the PCS and LASTXFER fields in the MSBs. However, the SPI still controls the number of bits (8 to 16) to be transferred through MISO and MOSI lines with the chip select configuration registers. This is not the optimal means in terms of memory size for the buffers, but it provides a very effective means to exchange data with several peripherals without any intervention of the processor.

#### 48.7.3.7 Peripheral Chip Select Decoding

The user can program the SPI to operate with up to 15 slave peripherals by decoding the four chip select lines, NPCS0 to NPCS3 with an external decoder/demultiplexer (see figure below). This can be enabled by setting SPI_MR.PCSDEC.

When operating without decoding, the SPI makes sure that in any case only one chip select line is activated, i.e., one NPCS line driven low at a time. If two bits are defined low in a PCS field, only the lowest numbered chip select is driven low.

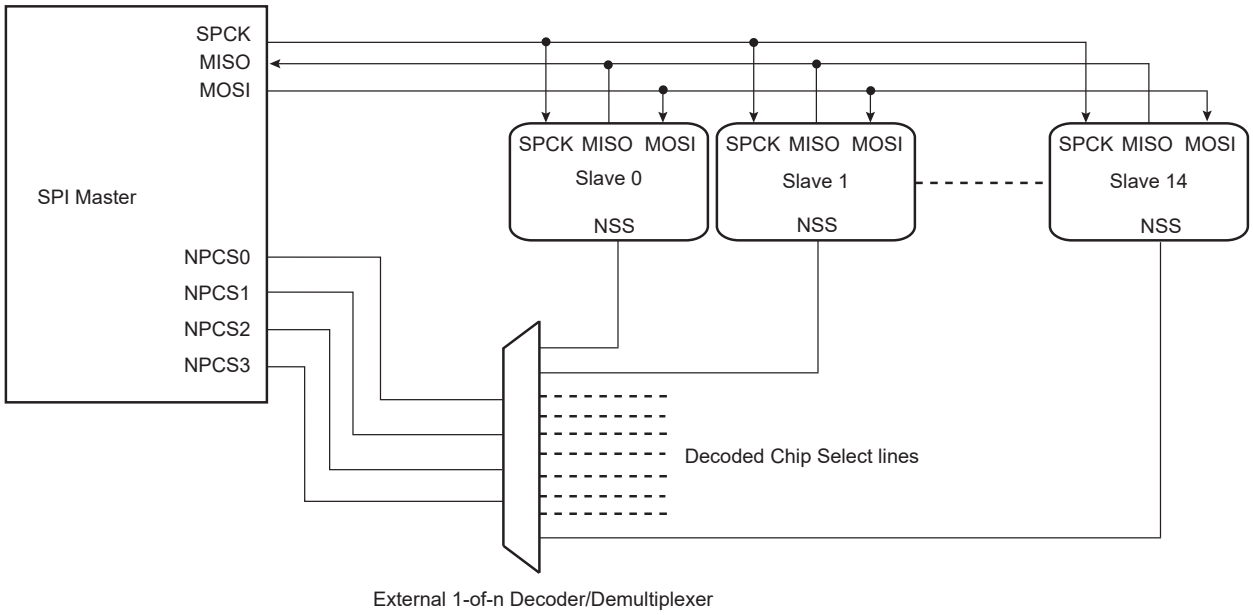
When operating with decoding, the SPI directly outputs the value defined by the PCS field on the NPCS lines of either SPI_MR or SPI_TDR (depending on PS).

As the SPI sets a default value of 0xF on the chip select lines (i.e., all chip select lines at 1) when not processing any transfer, only 15 peripherals can be decoded.

The SPI has four chip select registers (SPI_CSR0...SPI_CSR3). As a result, when external decoding is activated, each NPCS chip select defines the characteristics of up to four peripherals. As an example, SPI_CRS0 defines the characteristics of the externally decoded peripherals 0 to 3, corresponding to the PCS values 0x0 to 0x3. Consequently, the user has to make sure to connect compatible peripherals on the decoded chip select lines 0 to 3, 4 to 7, 8 to 11 and 12 to 14. The following figure shows this type of implementation.

If SPI_CSRx.CSAAT bit is used, with or without the DMAC, the Mode Fault detection for NPCS0 line must be disabled. This is not needed for all other chip select lines since Mode Fault detection is only on NPCS0.

**Figure 48-10. Chip Select Decoding Application Block Diagram: Single Master/Multiple Slave Implementation**




#### 48.7.3.8 Peripheral Deselection without DMA

During a transfer of more than one unit of data on a chip select without the DMA, SPI_TDR is loaded by the processor, the TDRE flag rises as soon as the content of SPI_TDR is transferred into the internal shift register. When this flag is detected high, SPI_TDR can be reloaded. If this reload by the processor occurs before the end of the current transfer and if the next transfer is performed on the same chip select as the current transfer, the chip select is not deasserted between the two transfers. But depending on the application software handling the SPI status register flags (by interrupt or polling method) or servicing other interrupts or other tasks, the processor may not reload SPI_TDR in time to keep the chip select active (low). A null DLYBCT value (delay between consecutive transfers) in SPI_CSR, gives even less time for the processor to reload SPI_TDR. With some SPI slave peripherals, if the chip select line must remain active (low) during a full set of transfers, communication errors can occur.

To facilitate interfacing with such devices, the chip select registers [SPI_CSR0...SPI_CSR3] can be programmed with the Chip Select Active After Transfer (CSAAT) bit at 1. This allows the chip select lines to remain in their current state (low = active) until a transfer to another chip select is required. Even if SPI_TDR is not reloaded, the chip select remains active. To deassert the chip select line at the end of the transfer, the Last Transfer (LASTXFER) bit in SPI_CR must be set after writing the last data to transmit into SPI_TDR.

#### 48.7.3.9 Peripheral Deselection with DMA

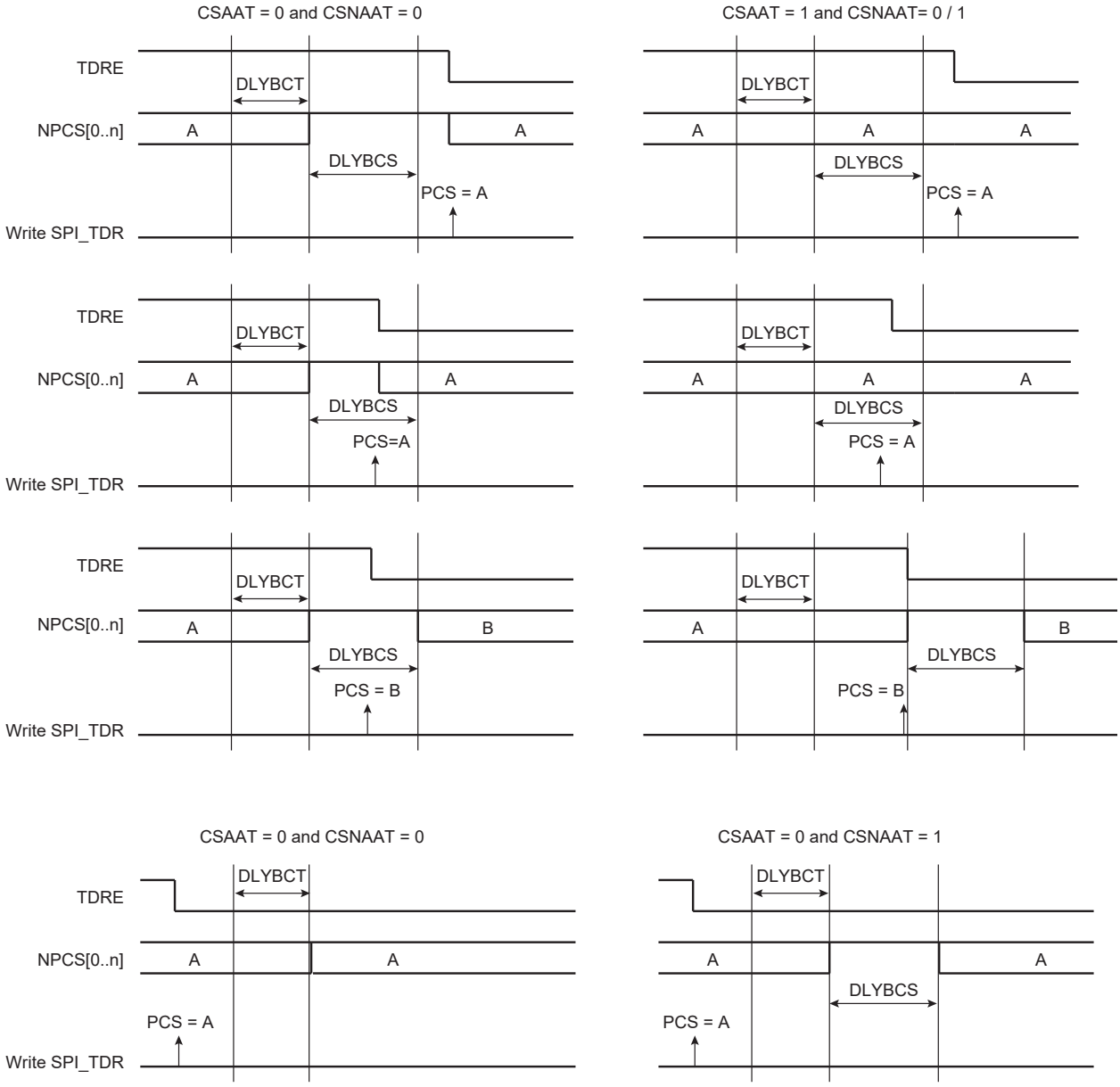
DMA provides faster reloads of SPI_TDR compared to software. However, depending on the system activity, it is not guaranteed that SPI_TDR is written with the next data before the end of the current transfer. Consequently, data can be lost by the deassertion of the NPCS line for SPI slave peripherals requiring the chip select line to remain active between two transfers. The only way to guarantee a safe transfer in this case is the use of the CSAAT and LASTXFER bits.

When the CSAAT bit is configured to 0, the NPCS does not rise in all cases between two transfers on the same peripheral. During a transfer on a chip select, the TDRE flag rises as soon as the content of SPI_TDR is transferred into the internal shift register. When this flag is detected, SPI_TDR can be reloaded. If this reload occurs before the end of the current transfer and if the next transfer is performed on the same chip select as the current transfer, the chip select is not deasserted between the two

transfers. This can lead to difficulties to interface with some serial peripherals requiring the chip select to be deasserted after each transfer. To facilitate interfacing with such devices, SPI_CSR can be programmed with the Chip Select Not Active After Transfer (CSNAAT) bit at 1. This allows the chip select lines to be deasserted systematically during a time “DLYBCS” (the value of the CSNAAT bit is processed only if the CSAAT bit is configured to 0 for the same chip select).

The following figure shows different peripheral deselection cases and the effect of the CSAAT and CSNAAT bits.

**Figure 48-11. Peripheral Deselection**




### 48.7.3.10 Mode Fault Detection

The SPI has the capability to operate in multimaster environment. Consequently, the NPCS0/NSS line must be monitored. If one of the masters on the SPI bus is currently transmitting, the NPCS0/NSS line is low and the SPI must not transmit any data. A mode fault is detected when the SPI is programmed in Master mode and a low level is driven by an external master on the NPCS0/NSS signal. In multimaster

environment, NPCS0, MOSI, MISO and SPCK pins must be configured in open drain (through the PIO controller). When a mode fault is detected, SPI_SR.MODF bit is set until SPI_SR is read and the SPI is automatically disabled until it is reenabled by setting SPI_CR.SPIEN bit.

By default, the mode fault detection is enabled. The user can disable it by setting SPI_MR.MODFDIS bit.

### 48.7.4 SPI Slave Mode

When operating in Slave mode, the SPI processes data bits on the clock provided on the SPI clock pin (SPCK).

The SPI waits until NSS goes active before receiving the serial clock from an external master. When NSS falls, the clock is validated and the data is loaded in SPI_RDR depending on the configuration of SPI_CSR0.BITS. These bits are processed following a phase and a polarity defined respectively by the NCPHA and CPOL bits in SPI_CSR0. Note that the fields BITS, CPOL and NCPHA of the other chip select registers (SPI_CSR1...SPI_CSR3) have no effect when the SPI is programmed in Slave mode.

The bits are shifted out on the MISO line and sampled on the MOSI line.

**Note:** For more information on SPI_CSRx.BITS, see the note in section SPI Chip Select Register.

When all bits are processed, the received data is transferred in SPI_RDR and the RDRF bit rises. If SPI_RDR has not been read before new data is received, the Overrun Error Status (OVRES) bit in SPI_SR is set. As long as this flag is set, data is loaded in SPI_RDR. The user must read SPI_SR to clear the OVRES bit.

When a transfer starts, the data shifted out is the data present in the internal shift register. If no data has been written in SPI_TDR, the last data received is transferred. If no data has been received since the last reset, all bits are transmitted low, as the internal shift register resets to 0.

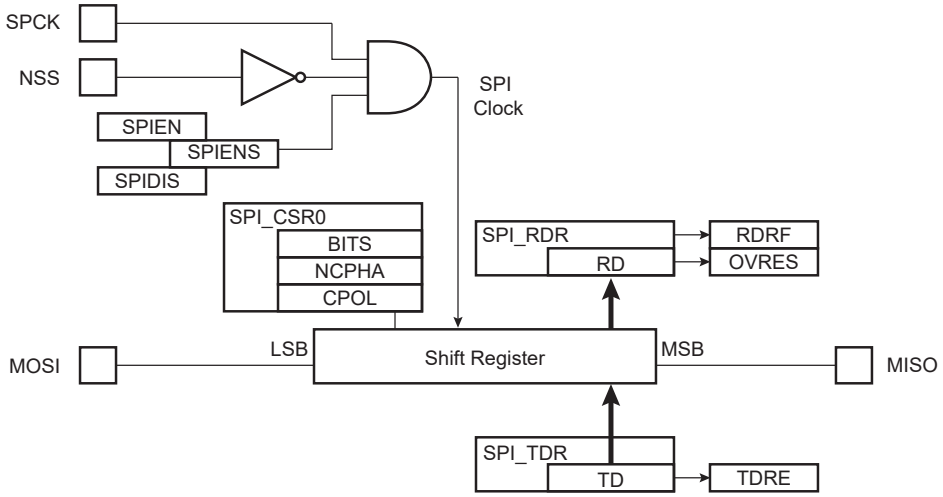
When a first data is written in SPI_TDR, it is transferred immediately in the internal shift register and the TDRE flag rises. If new data is written, it remains in SPI_TDR until a transfer occurs, i.e., NSS falls and there is a valid clock on the SPCK pin. When the transfer occurs, the last data written in SPI_TDR is transferred in the internal shift register and the TDRE flag rises. This enables frequent updates of critical variables with single transfers.

Then, new data is loaded in the internal shift register from SPI_TDR. If no character is ready to be transmitted, i.e., no character has been written in SPI_TDR since the last load from SPI_TDR to the internal shift register, SPI_TDR is retransmitted. In this case the Underrun Error Status Flag (UNDES) is set in SPI_SR.

If NSS rises between two characters, it must be kept high for two MCK clock periods or more and the next SPCK capture edge must not occur less than four MCK periods after NSS rise.

The following figure shows a block diagram of the SPI when operating in Slave mode.

**Figure 48-12. Slave Mode Functional Block Diagram**




### 48.7.5 SPI Comparison Function on Received Character

The comparison is only relevant for SPI Slave mode (SPI_MR.MSTR=0).

The effect of a comparison match changes if the system is in Wait or Active mode.

In Wait mode, if asynchronous partial wakeup is enabled, a system wakeup is performed (see section SPI Asynchronous and Partial Wakeup (SleepWalking)).

In Active mode, the CMP flag in SPI_SR is raised. It is set when the received character matches the conditions programmed in the SPI Comparison Register (SPI_CMPR). The CMP flag is set as soon as SPI_RDR is loaded with the new received character. The CMP flag is cleared by reading SPI_SR.

SPI_CMPR (see section SPI Comparison Register) can be programmed to provide different comparison methods. These are listed below:

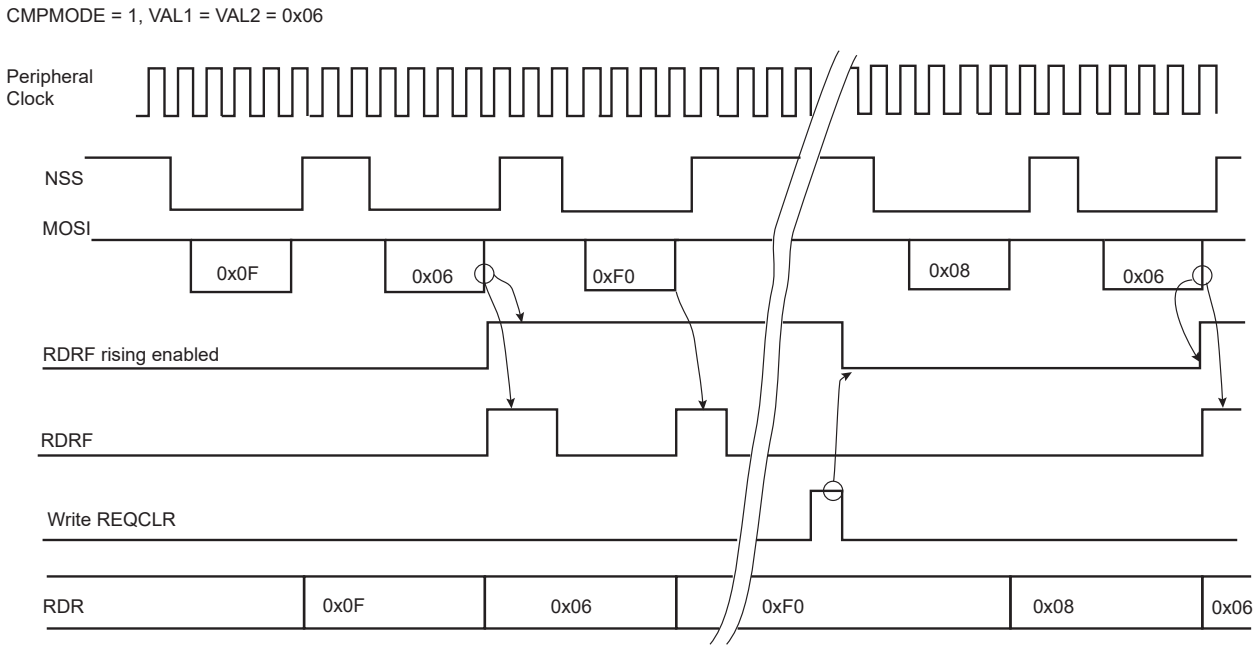
- If VAL1 equals VAL2, then the comparison is performed on a single value and the flag is set to 1 if the received character equals VAL1.
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 sets the CMP flag.
- If VAL1 is strictly higher than VAL2, then the flag CMP is set to 1 if any received character equals VAL1 or VAL2.

When SPI_MR.CMPMODE is cleared, all received data is loaded in SPI_RDR and the CMP flag provides the status of the comparison result.

By setting SPI_MR.CMPMODE, the comparison result triggers the start of SPI_RDR loading (see the following figure). The trigger condition exists as soon as the received character value matches the conditions defined by VAL1 and VAL2 in SPI_CMPR. The comparison trigger event is restarted by setting SPI_CR.REQCLR if SleepWalking is disabled.

The value programmed in VAL1 and VAL2 fields must not exceed the maximum value of the received character (see SPI_CSR0.BITS).

**Figure 48-13. Receive Data Register Management**




### 48.7.6 SPI Asynchronous and Partial Wakeup (SleepWalking)

This operating mode is a means of data preprocessing that qualifies an incoming event, thus allowing the SPI to decide whether or not to wake up the system. Asynchronous and partial wakeup is mainly used when the system is in Wait mode (refer to the section “Power Management Controller (PMC)” for further details). It can also be enabled when the system is fully running.

Asynchronous and partial wakeup can be used only when SPI is configured in Slave mode (SPI_MR.MSTR is cleared).

The maximum SPI clock (SPCK) frequency that can be provided by the SPI master is bounded by the peripheral clock frequency. The SPCK frequency must be lower than or equal to the peripheral clock. The NSS line must be deasserted by the SPI master between two characters. The NSS deassertion duration time must be greater than or equal to six peripheral clock periods. The time between the assertion of NSS line (falling edge) and the first edge of the SPI clock must be higher than 5 µs.

SPI_RDR must be read before enabling the asynchronous and partial wakeup.

When asynchronous and partial wakeup is enabled for the SPI (refer to the section “Power Management Controller (PMC)”), the PMC decodes a clock request from the SPI. The request is generated as soon as there is a falling edge on the NSS line as this may indicate the beginning of a frame. If the system is in Wait mode (processor and peripheral clocks switched off), the PMC restarts the fast RC oscillator and provides the clock only to the SPI.

The SPI processes the received frame and compares the received character with SPI_CMPR.VAL1 and SPI_CMPR.VAL2 (see section SPI Comparison Register).

The SPI instructs the PMC to disable the peripheral clock if the received character value does not meet the conditions defined by SPI_CMPR.VAL1 and SPI_CMPR.VAL2 (see figure Asynchronous Event Generating Only Partial Wakeup).

If the received character value meets the conditions, the SPI instructs the PMC to exit the system from Wait mode (see figure Asynchronous Wakeup Use Case Example).

The VAL1 and VAL2 fields can be programmed to provide different comparison methods and thus matching conditions.

- If VAL1 = VAL2, then the comparison is performed on a single value and the wakeup is triggered if the received character equals VAL1.
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 wakes up the system.
- If VAL1 is strictly higher than VAL2, the wakeup is triggered if any received character equals VAL1 or VAL2.
- If VAL1 = 0 and VAL2 = 65535, the wakeup is triggered as soon as a character is received.

If the processor and peripherals are running, the SPI can be configured in Asynchronous and Partial Wakeup mode by enabling the PMC_SLPWK_ER (refer to the section “Power Management Controller (PMC)”). When activity is detected on the receive line, the SPI requests the clock from the PMC and the comparison is performed. If there is a comparison match, the SPI continues to request the clock. If there is no match, the clock is switched off for the SPI only, until a new activity is detected.

The CMPMODE configuration has no effect when Asynchronous and Partial Wakeup mode is enabled for the SPI (refer to PMC_SLPWK_ER in the section “Power Management Controller (PMC)”).

When the system is in Active mode and the SPI enters Asynchronous and Partial Wakeup mode, the flag RDRF must be programmed as the unique source of the SPI interrupt.

When the system exits Wait mode as the result of a matching condition, the RDRF flag is used to determine if the SPI is the source for the exit from Wait mode.

**Figure 48-14. Asynchronous Wakeup Use Case Example**

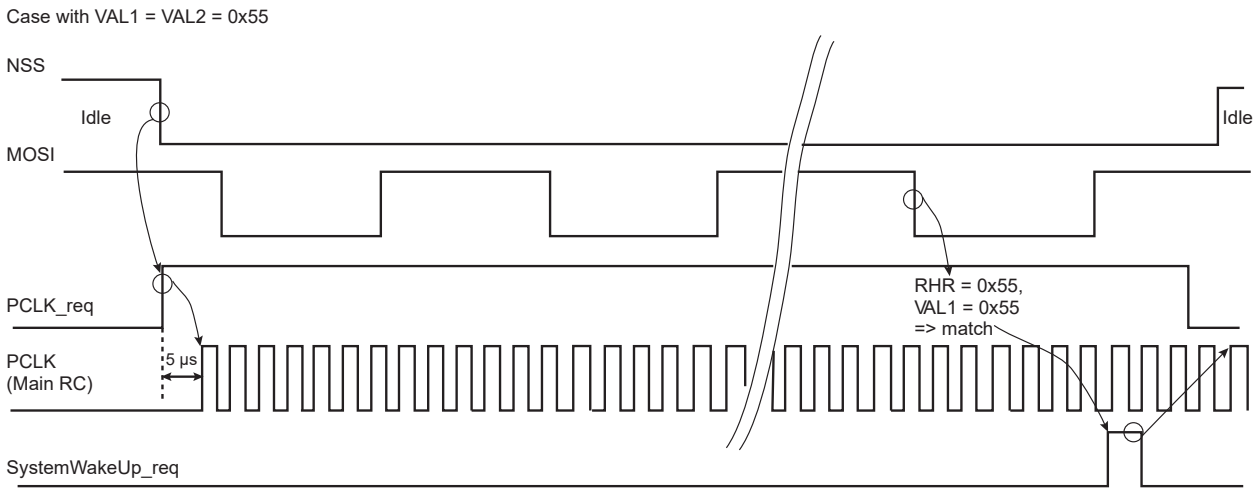
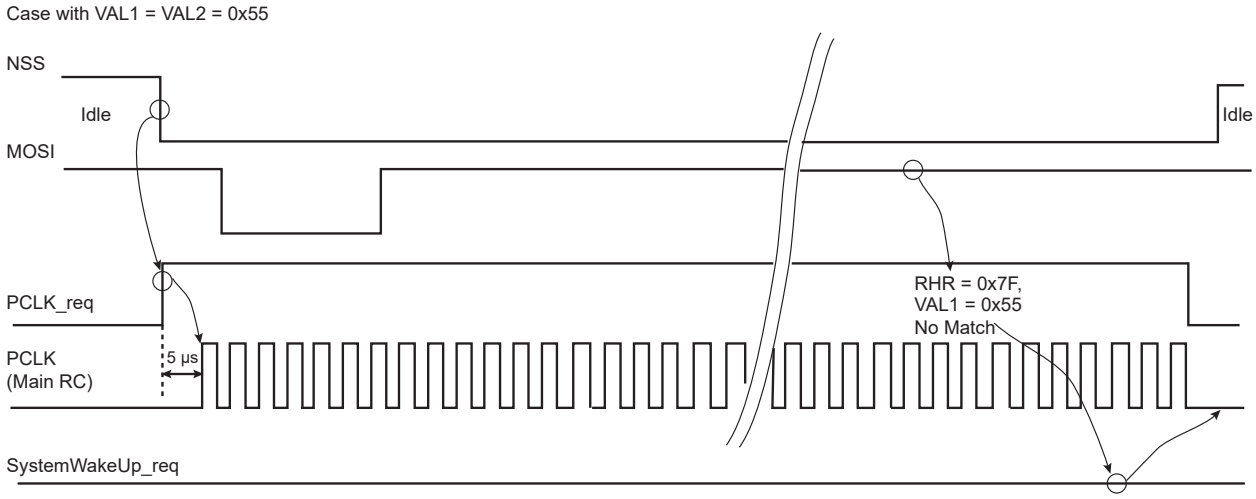


**Figure 48-15. Asynchronous Event Generating Only Partial Wakeup**




### 48.7.7 FIFOs

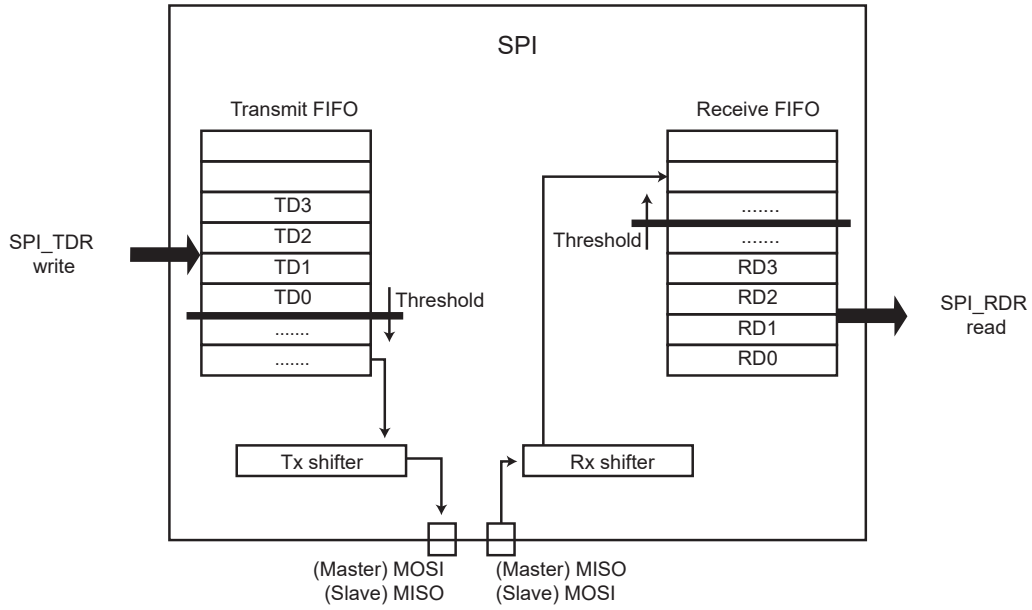
#### 48.7.7.1 Overview

The SPI includes two FIFOs which can be enabled/disabled using SPI_CR.FIFOEN/FIFODIS. The SPI must be disabled (SPI_CR.SPIDIS) before enabling or disabling the SPI FIFOs.

Writing SPI_CR.FIFOEN to '1' enables a 16-data Transmit FIFO and a 16-data Receive FIFO.

It is possible to write or to read single or multiple data in the same access to SPI_TDR/RDR. Refer to sections Single Data Mode and Multiple Data Mode.

**Figure 48-16. FIFOs Block Diagram**




#### 48.7.7.2 Sending Data with FIFO Enabled

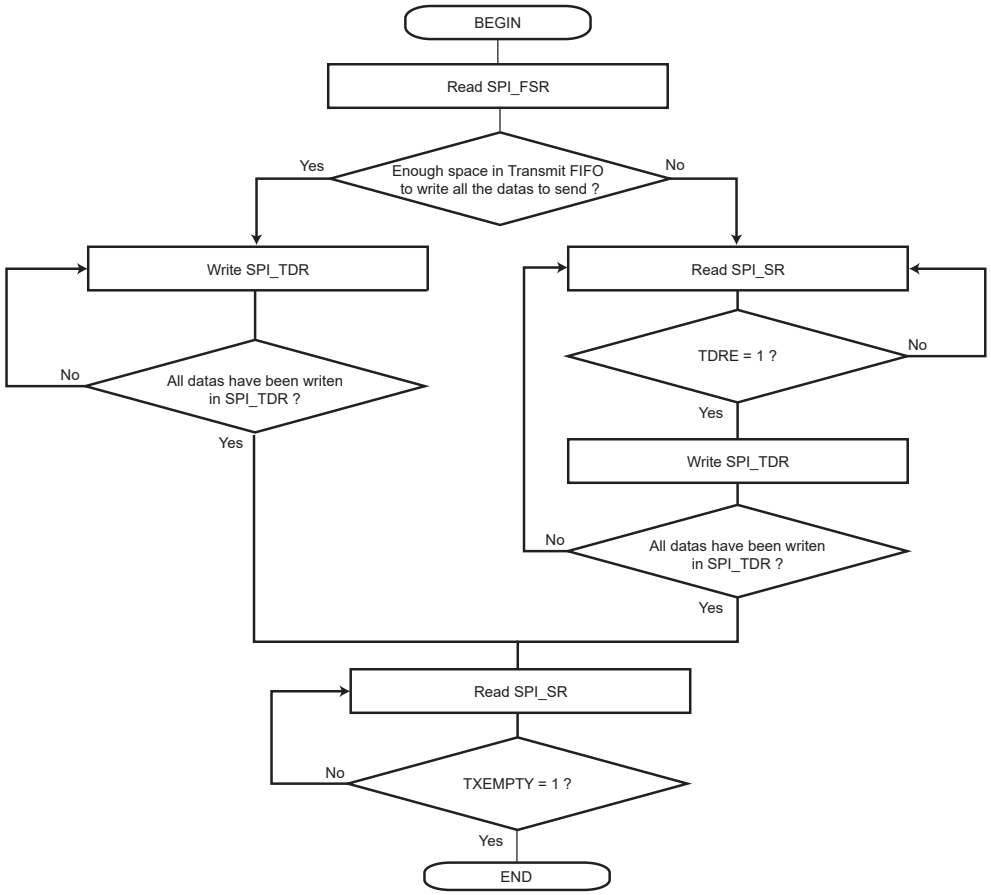
When the Transmit FIFO is enabled, write access to SPI_TDR loads the Transmit FIFO.

The FIFO level is provided in SPI_FLR.TXFL. If the FIFO can accept the number of data to be transmitted, there is no need to monitor SPI_SR.TDRE and the data can be successively written in SPI_TDR.

If the FIFO cannot accept the data due to insufficient space, wait for the TDRE flag to be set before writing the data in SPI_TDR.

When the space in the FIFO allows only a portion of the data to be written, the TDRE flag must be monitored before writing the remaining data.

**Figure 48-17. Sending Data with FIFO**

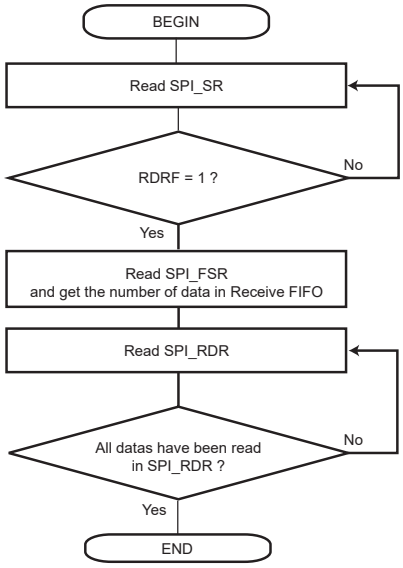



#### 48.7.7.3 Receiving Data with FIFO Enabled

When the Receive FIFO is enabled, SPI_RDR access reads the FIFO.

When data are present in the Receive FIFO (RDRF flag set to ‘1’), the exact number of data can be checked with SPI_FLR.RXFL. All the data can be read successively in SPI_RDR without checking the RDRF flag between each access.

**Figure 48-18. Receiving Data with FIFO**




#### 48.7.7.4 Clearing/Flushing FIFOs

Each FIFO can be cleared/flushed using SPI_CR.TXFCLR/RXFCLR.

#### 48.7.7.5 TXEMPTY, TDRE and RDRF Behavior

SPI_SR.TXEMPTY, SPI_SR.TDRE and SPI_SR.RDRF flags display a specific behavior when FIFOs are enabled.

The TXEMPTY flag is cleared as long as there are characters in the Transmit FIFO or in the internal shift register. TXEMPTY is set when there are no characters in the Transmit FIFO and in the internal shift register.

TDRE indicates if a data can be written in the Transmit FIFO. Thus the TDRE flag is set as long as the Transmit FIFO can accept new data. Refer to figure TDRE in Single Data Mode and TXRDYM=0.

RDRF indicates if an unread data is present in the Receive FIFO. Thus the RDRF flag is set as soon as one unread data is in the Receive FIFO. Refer to figure RDRF in Single Data Mode and RXRDYM=0.

TDRE and RDRF behavior can be modified using the TXRDYM and RXRDYM fields in the SPI FIFO Mode Register (SPI_FMR) to reduce the number of accesses to SPI_TDR/RDR. However, for some configurations, the following constraints apply:

- When the Variable Peripheral Select mode is used (SPI_MR.PS=1), SPI_FMR.TXRDYM/RXRDYM must be cleared.
- In Master mode (SPI_MR.MSTR=1), SPI_FMR.RXRDYM must be cleared.

As an example, in Master mode, the Transmit FIFO can be loaded with multiple data in the same access by configuring TXRDYM>0.

See section SPI FIFO Mode Register for the FIFO configuration.

**Figure 48-19. TDRE in Single Data Mode and TXRDYM=0**

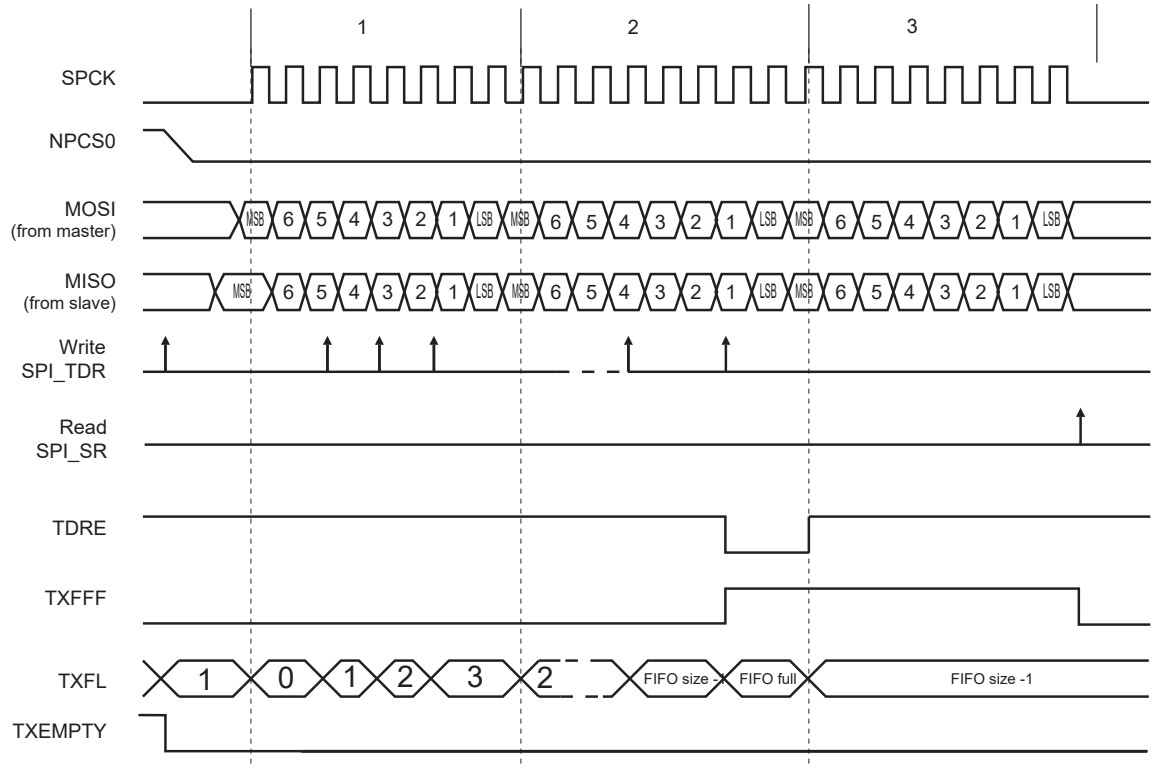
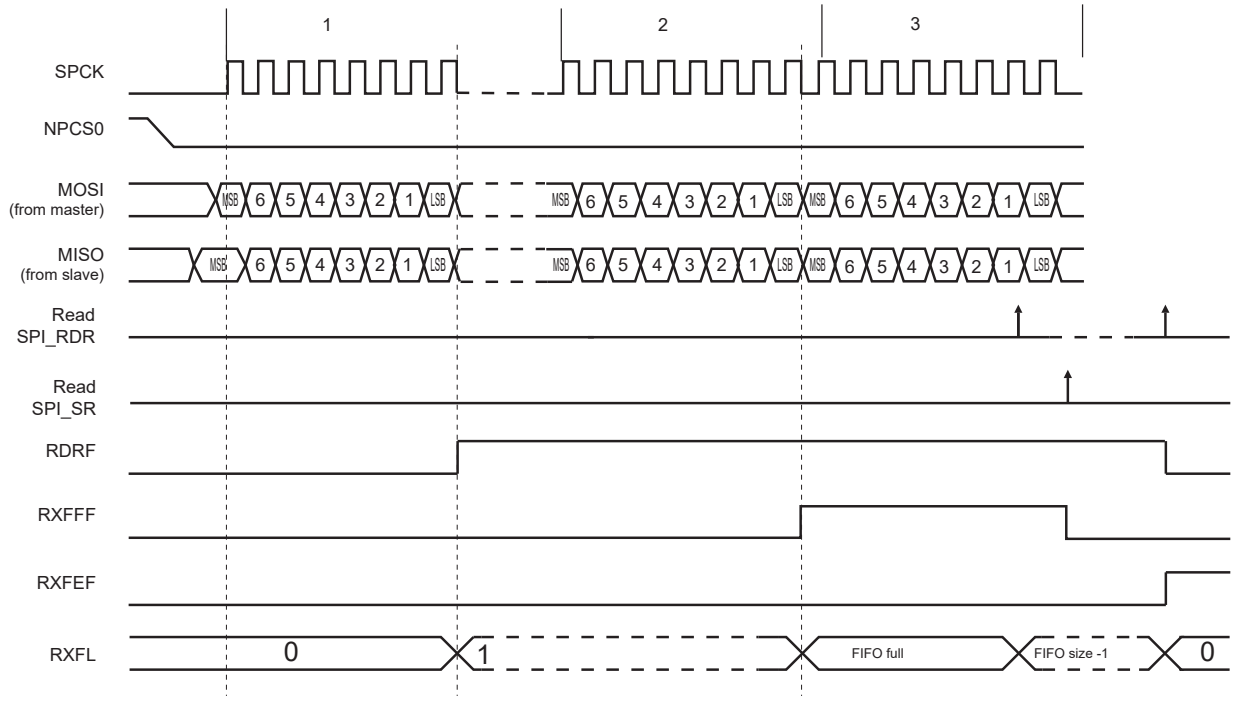



**Figure 48-20. RDRF in Single Data Mode and RXRDYM=0**




### 48.7.7.6 Single Data Mode

In Single Data mode, only one data is written every time SPI_TDR is accessed, and only one data is read every time SPI_RDR is accessed.

When SPI_FMR.TXRDYM = 0, the Transmit FIFO operates in Single Data mode.

When SPI_FMR.RXRDYM = 0, the Receive FIFO operates in Single Data mode.

If Master mode is used (SPI_MR.MSTR=1), the Receive FIFO must operate in Single Data mode.

If Variable Peripheral Select mode is used (SPI_MR.PS=1), the Transmit FIFO must operate in Single Data mode.

See sections SPI Transmit Data Register and SPI Receive Data Register.

#### 48.7.7.6.1 DMAC

When FIFOs operate in Single Data mode, the DMAC transfer type must be configured either in bytes, halfwords or words depending on SPI_MR.PS bit value and SPI_CSRx.BITS field value.

The same conditions for transfer type apply when FIFOs are disabled.

### 48.7.7.7 Multiple Data Mode

Multiple Data mode minimizes the number of accesses by concatenating the data to send/read in one access.

When SPI_FMR.TXRDYM > 0, the Transmit FIFO operates in Multiple Data mode.

When SPI_FMR.RXRDYM > 0, the Receive FIFO operates in Multiple Data mode.

Multiple data can be read from the Receive FIFO only in Slave mode (SPI_MR.MSTR=0).

The Transmit FIFO can be loaded with multiple data in the same access by configuring TXRDYM>0 and when SPI_MR.PS=0.

In Multiple Data mode, up to two data can be written in one SPI_TDR write access. It is also possible to read up to four data in one SPI_RDR access if SPI_CSRx.BITS is configured to ‘0’ (8-bit data size) and up to two data if SPI_CSRx.BITS is configured to a value other than ‘0’ (more than 8-bit data size).

The number of data to write/read is defined by the size of the register access. If the access is a byte-size register access, only one data is written/read. If the access is a halfword size register access, then up to two data are read and only one data is written. Lastly, if the access is a word-size register access, then up to four data are read and up to two data are written.

Written/read data are always right-aligned, as described in sections SPI Receive Data Register (FIFO Multiple Data, 8-bit), SPI Receive Data Register (FIFO Mutliple Data, 16-bit) and SPI Transmit Data Register (FIFO Multiple Data, 8 to 16-bit).

As an example, if the Transmit FIFO is empty and there are six data to send, either of the following write accesses may be performed:

- six SPI_TDR-byte write accesses
- three SPI_TDR-halfword write accesses

With a Receive FIFO containing six data, any of the following read accesses may be performed:

- six SPI_RDR-byte read accesses
- three SPI_RDR-halfword read accesses
- one SPI_RDR-word read access and one SPI_RDR-halfword read access

#### 48.7.7.7.1 TDRE and RDRF Configuration

In Multiple Data mode, it is possible to write one or more data in the same SPI_TDR/SPI_RDR access. The TDRE flag indicates if one or more data can be written in the FIFO depending on the configuration of SPI_FMR.TXRDYM/RXRDYM.

As an example, if two data are written each time in SPI_TDR, it is useful to configure the TXRDYM field to the value ‘1’ so that the TDRE flag is at ‘1’ only when at least two data can be written in the Transmit FIFO.

Similarly, if four data are read each time in SPI_RDR, it is useful to configure the RXRDYM field to the value ‘2’ so that the RDRF flag is at ‘1’ only when at least four unread data are in the Receive FIFO.

#### 48.7.7.7.2 DMAC

It is mandatory to configure DMAC channel size (byte, halfword or word) according to FLEX_SPI_FMR.TXRDYM/RXRDYM configuration. See section Multiple Data Mode for constraints.

### 48.7.7.8 FIFO Pointer Error

A FIFO overflow is reported in SPI_SR.

If the Transmit FIFO is full and a write access is performed on SPI_TDR, it generates a Transmit FIFO pointer error and sets SPI_SR.TXFPTEF.

In Multiple Data mode, if the number of data written in SPI_TDR (according to the register access size) is greater than the free space in the Transmit FIFO, a Transmit FIFO pointer error is generated and SPI_SR.TXFPTEF is set.

A FIFO underflow is reported in SPI_SR.

In Multiple Data mode, if the number of data read in SPI_RDR (according to the register access size) is greater than the number of unread data in the Receive FIFO, a Receive FIFO pointer error is generated and SPI_SR.RXFPTEF is set.

No pointer error occurs if the FIFO state/level is checked before writing/reading in SPI_TDR/SPI_RDR. The FIFO state/level can be checked either with TXRDY, RXRDY, TXFL or RXFL. When a pointer error occurs, other FIFO flags may not behave as expected; their states should be ignored.

If a pointer error occurs, a software reset must be performed using SPI_CR.SWRST (configuration will be lost).

### 48.7.7.9 FIFO Thresholds

Each Transmit and Receive FIFO includes a threshold feature used to set a flag and an interrupt when a FIFO threshold is crossed. Thresholds are defined as a number of data in the FIFO, and the FIFO state (TXFL or RXFL) represents the number of data currently in the FIFO.

The Transmit FIFO threshold can be set using SPI_FMR.TXFTHRES. Each time the Transmit FIFO goes from the ‘above threshold’ to the ‘equal or below threshold’ state, SPI_SR.TXFTHF is set. The application is warned that the Transmit FIFO has reached the defined threshold and that it can be reloaded.

The Receive FIFO threshold can be set using SPI_FMR.RXFTHRES. Each time the Receive FIFO goes from the ‘below threshold’ to the ‘equal or above threshold’ state, SPI_SR.RXFTHF is set. The application is warned that the Receive FIFO has reached the defined threshold and that it can be read to prevent an underflow.

The TXFTHF and RXFTHF flags can be configured to generate an interrupt using SPI_IER and SPI_IDR.

### 48.7.7.10 FIFO Flags

FIFOs come with a set of flags which can be configured to generate interrupts through SPI_IER and SPI_IDR.

FIFO flags state can be read in SPI_SR. They are cleared when SPI_SR is read.

### 48.7.8 Register Write Protection

To prevent any single software error from corrupting SPI behavior, certain registers in the address space can be write-protected in the SPI Write Protection Mode Register (SPI_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the SPI Write Protection Status Register (SPI_WPSR) is set and the WPVSRC field indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading SPI_WPSR.

The following registers are write-protected when WPEN is set in SPI_WPMR:

- SPI Mode Register
- SPI Chip Select Register

## 48.8 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | SPI_CR | 7:0 | SWRST | | | | | | SPIDIS | SPIEN |
| | | 15:8 | | | | REQCLR | | | | |
| | | 23:16 | | | | | | | RXFCLR | TXFCLR |
| | | 31:24 | FIFODIS | FIFOEN | | | | | | LASTXFER |
| 0x04 | SPI_MR | 7:0 | LLB | | WDRBT | MODFDIS | BRSRCCLK | PCSDEC | PS | MSTR |
| | | 15:8 | | | | CMPMODE | | | | LSBHALF |
| | | 23:16 | | | | | PCS[3:0] | | | |
| | | 31:24 | DLYBCS[7:0] | | | | | | | |
| 0x08 | SPI_RDR | 7:0 | RD[7:0] | | | | | | | |
| | | 15:8 | RD[15:8] | | | | | | | |
| | | 23:16 | | | | | PCS[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | SPI_RDR (FIFO_MULTI_DATA_8) | 7:0 | RD0[7:0] | | | | | | | |
| | | 15:8 | RD1[7:0] | | | | | | | |
| | | 23:16 | RD2[7:0] | | | | | | | |
| | | 31:24 | RD3[7:0] | | | | | | | |
| 0x08 | SPI_RDR (FIFO_MULTI_DATA_16) | 7:0 | RD0[7:0] | | | | | | | |
| | | 15:8 | RD0[15:8] | | | | | | | |
| | | 23:16 | RD1[7:0] | | | | | | | |
| | | 31:24 | RD1[15:8] | | | | | | | |
| 0x0C | SPI_TDR | 7:0 | TD[7:0] | | | | | | | |
| | | 15:8 | TD[15:8] | | | | | | | |
| | | 23:16 | | | | | PCS[3:0] | | | |
| | | 31:24 | | | | | | | | LASTXFER |
| 0x0C | SPI_TDR (FIFO_MULTI_DATA) | 7:0 | TD0[7:0] | | | | | | | |
| | | 15:8 | TD0[15:8] | | | | | | | |
| | | 23:16 | TD1[7:0] | | | | | | | |
| | | 31:24 | TD1[15:8] | | | | | | | |
| 0x10 | SPI_SR | 7:0 | | | | | OVRES | MODF | TDRE | RDRF |
| | | 15:8 | | | | | CMP | UNDES | TXEMPTY | NSSR |
| | | 23:16 | | | | | | | | SPIENS |
| | | 31:24 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| 0x14 | SPI_IER | 7:0 | | | | | OVRES | MODF | TDRE | RDRF |
| | | 15:8 | | | | | CMP | UNDES | TXEMPTY | NSSR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| 0x18 | SPI_IDR | 7:0 | | | | | OVRES | MODF | TDRE | RDRF |
| | | 15:8 | | | | | CMP | UNDES | TXEMPTY | NSSR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x1C | SPI_IMR | 7:0 | | | | | OVRES | MODF | TDRE | RDRF |
| | | 15:8 | | | | | CMP | UNDES | TXEMPTY | NSSR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| 0x20 ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | SPI_CSR0 | 7:0 | BITS[3:0] | | | | CSAAT | CSNAAT | NCPHA | CPOL |
| | | 15:8 | SCBR[7:0] | | | | | | | |
| | | 23:16 | DLYBS[7:0] | | | | | | | |
| | | 31:24 | DLYBCT[7:0] | | | | | | | |
| 0x34 | SPI_CSR1 | 7:0 | BITS[3:0] | | | | CSAAT | CSNAAT | NCPHA | CPOL |
| | | 15:8 | SCBR[7:0] | | | | | | | |
| | | 23:16 | DLYBS[7:0] | | | | | | | |
| | | 31:24 | DLYBCT[7:0] | | | | | | | |
| 0x38 | SPI_CSR2 | 7:0 | BITS[3:0] | | | | CSAAT | CSNAAT | NCPHA | CPOL |
| | | 15:8 | SCBR[7:0] | | | | | | | |
| | | 23:16 | DLYBS[7:0] | | | | | | | |
| | | 31:24 | DLYBCT[7:0] | | | | | | | |
| 0x3C | SPI_CSR3 | 7:0 | BITS[3:0] | | | | CSAAT | CSNAAT | NCPHA | CPOL |
| | | 15:8 | SCBR[7:0] | | | | | | | |
| | | 23:16 | DLYBS[7:0] | | | | | | | |
| | | 31:24 | DLYBCT[7:0] | | | | | | | |
| 0x40 | SPI_FMR | 7:0 | | | RXRDYM[1:0] | | | | TXRDYM[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | TXFTHRES[5:0] | | | | | |
| | | 31:24 | | | RXFTHRES[5:0] | | | | | |
| 0x44 | SPI_FLR | 7:0 | | | TXFL[5:0] | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | RXFL[5:0] | | | | | |
| | | 31:24 | | | | | | | | |
| 0x48 | SPI_CMPR | 7:0 | VAL1[7:0] | | | | | | | |
| | | 15:8 | VAL1[15:8] | | | | | | | |
| | | 23:16 | VAL2[7:0] | | | | | | | |
| | | 31:24 | VAL2[15:8] | | | | | | | |
| 0x4C ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | SPI_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xE8 | SPI_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 48.8.1 SPI Control Register

**Name:** SPI_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FIFODIS | FIFOEN | | | | | | LASTXFER |
| Access | W | W | | | | | | W |
| Reset | – | – | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXFCLR | TXFCLR |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | REQCLR | | | | |
| Access | | | | W | | | | |
| Reset | | | | – | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWRST | | | | | | SPIDIS | SPIEN |
| Access | W | | | | | | W | W |
| Reset | – | | | | | | – | – |

**Bit 31 – FIFODIS** FIFO Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Transmit and Receive FIFOs. |

**Bit 30 – FIFOEN** FIFO Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Transmit and Receive FIFOs. |

**Bit 24 – LASTXFER** Last Transfer
Refer to section Peripheral Selection for more details.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The current NPCS is deasserted after the character written in TD has been transferred. When SPI_CSRx.CSAAT is set, the communication with the current serial peripheral can be closed by raising the corresponding NPCS line as soon as TD transfer is completed. |

**Bit 17 – RXFCLR** Receive FIFO Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Empties the Receive FIFO. |

**Bit 16 – TXFCLR** Transmit FIFO Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Empties the Transmit FIFO. |

**Bit 12 – REQCLR** Request to Clear the Comparison Trigger
SleepWalking enabled:

0: No effect.

1: Clears the potential clock request currently issued by SPI, thus the potential system wakeup is cancelled.

SleepWalking disabled:

0: No effect.

1: Restarts the comparison trigger to enable SPI_RDR loading.

**Bit 7 – SWRST** SPI Software Reset
The SPI is in Slave mode after software reset.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Reset the SPI. A software-triggered hardware reset of the SPI interface is performed. |

**Bit 1 – SPIDIS** SPI Disable
All pins are set in Input mode after completion of the transmission in progress, if any.

If a transfer is in progress when SPIDIS is set, the SPI completes the transmission of the shifter register and does not start any new transfer, even if SPI_THR is loaded.

If both SPIEN and SPIDIS are equal to one when SPI_CR is written, the SPI is disabled.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the SPI. |

**Bit 0 – SPIEN** SPI Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the SPI to transfer and receive data. |

### 48.8.2 SPI Mode Register

**Name:** SPI_MR
**Offset:** 0x04
**Reset:** 0x0
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in theSPI Write Protection Mode Register .

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DLYBCS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PCS[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | CMPMODE | | | | LSBHALF |
| Access | | | | R/W | | | | R/W |
| Reset | | | | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LLB | | WDRBT | MODFDIS | BRSRCCLK | PCSDEC | PS | MSTR |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – DLYBCS[7:0]** Delay Between Chip Selects
This field defines the delay between the inactivation and the activation of NPCS. The DLYBCS time guarantees nonoverlapping chip selects and solves bus contentions in case of peripherals having long data float times.

If DLYBCS is lower than 6, six peripheral clock periods are inserted by default.

Otherwise, the following equations determine the delay:

If BRSRCCLK = 0:

$$\text{Delay Between Chip Selects} = \frac{\text{DLYBCS}}{\text{f}_{\text{peripheral clock}}}$$

If BRSRCCLK = 1:

$$\text{Delay Between Chip Selects} = \frac{\text{DLYBCS}}{\text{f}_{\text{GCLK}}}$$

**Bits 19:16 – PCS[3:0]** Peripheral Chip Select
This field is only used if fixed peripheral select is active (PS = 0).

If SPI_MR.PCSDEC = 0:

PCS = xxx0 NPCS[3:0] = 1110

PCS = xx01 NPCS[3:0] = 1101

PCS = x011 NPCS[3:0] = 1011

PCS = 0111 NPCS[3:0] = 0111

PCS = 1111 forbidden (no peripheral is selected)

(x = don't care)

If SPI_MR.PCSDEC = 1:

NPCS[3:0] output signals = PCS.

**Bit 12 – CMPMODE** Comparison Mode

| Value | Name | Description |
|---|---|---|
| 0 | FLAG_ONLY | Any character is received and comparison function drives CMP flag. |
| 1 | START_CONDITION | Comparison condition must be met to start reception of all incoming characters until REQCLR is set. |

**Bit 8 – LSBHALF** LSB Timing Selection

| Value | Description |
|---|---|
| 0 | To be used only if SPI slave LSB timing is 100% compliant with SPI standard (LSB duration is a full bit time). This value gives the better margin for SPI slave response delay (less than 1 SPCK clock cycle). |
| 1 | To be selected if the SPI slave LSB timing does not behave as the SPI standard (not triggered by NPCS deassertion in mode), the slave response delay is limited to less than 1/2 SPCK cycle. |

**Bit 7 – LLB** Local Loopback Enable
LLB controls the local loopback on the data shift register for testing in Master mode only (MISO is internally connected on MOSI).

| Value | Description |
|---|---|
| 0 | Local loopback path disabled. |
| 1 | Local loopback path enabled. |

**Bit 5 – WDRBT** Wait Data Read Before Transfer

| Value | Description |
|---|---|
| 0 | No Effect. In Master mode, a transfer can be initiated regardless of SPI_RDR state. |
| 1 | In Master mode, a transfer can start only if SPI_RDR is empty, i.e., does not contain any unread data. This mode prevents overrun error in reception. |

**Bit 4 – MODFDIS** Mode Fault Detection

| Value | Description |
|---|---|
| 0 | Mode fault detection enabled |
| 1 | Mode fault detection disabled |

**Bit 3 – BRSRCCLK** Bit Rate Source Clock
If bit BRSRCCLK = 1, the SCBR field in SPI_CSRx must be programmed with a value greater than 1.

0 (PERIPH_CLK): The peripheral clock is the source clock for the bit rate generation.

1 (GCLK): PMC GCLK is the source clock for the bit rate generation, thus the bit rate can be independent of the core/peripheral clock.

**Bit 2 – PCSDEC** Chip Select Decode
When PCSDEC = 1, up to 15 chip select signals can be generated with the four NPCS lines using an external 4-bit to 16-bit decoder. The chip select registers define the characteristics of the 15 chip selects, with the following rules:

SPI_CSR0 defines peripheral chip select signals 0 to 3.

SPI_CSR1 defines peripheral chip select signals 4 to 7.

SPI_CSR2 defines peripheral chip select signals 8 to 11.

SPI_CSR3 defines peripheral chip select signals 12 to 14.

| Value | Description |
|---|---|
| 0 | The chip select lines are directly connected to a peripheral device. |
| 1 | The four NPCS chip select lines are connected to a 4-bit to 16-bit decoder. |

**Bit 1 – PS** Peripheral Select

| Value | Description |
|---|---|
| 0 | Fixed Peripheral Select |
| 1 | Variable Peripheral Select |

**Bit 0 – MSTR** Master/Slave Mode

| Value | Description |
|---|---|
| 0 | SPI is in Slave mode |
| 1 | SPI is in Master mode |

### 48.8.3 SPI Receive Data Register

**Name:** SPI_RDR
**Offset:** 0x08
**Reset:** 0x0
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PCS[3:0] | | | |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 19:16 – PCS[3:0]** Peripheral Chip Select
In Master mode only, these bits indicate the value on the NPCS pins at the end of a transfer. Otherwise, these bits are read as zero.

When using Variable Peripheral Select mode (PS = 1 in SPI_MR), it is mandatory to set SPI_MR.WDRBT bit if the PCS field must be processed in SPI_RDR.

**Bits 15:0 – RD[15:0]** Receive Data
Data received by the SPI Interface is stored in this register in a right-justified format. Unused bits are read as zero.

### 48.8.4 SPI Receive Data Register (FIFO Multiple Data, 8-bit)

**Name:** SPI_RDR (FIFO_MULTI_DATA_8)
**Offset:** 0x08
**Reset:** 0x0
**Property:** Read-only

If FIFO is enabled (FIFOEN bit in SPI_CR), refer to section Multiple Data Mode.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RD3[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RD2[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RD1[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RD0[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0:7, 8:15, 16:23, 24:31 – RDx** Receive Data
First unread data in the Receive FIFO. Data received by the SPI interface is stored in this register in a right-justified format. Unused bits are read as zero.

### 48.8.5 SPI Receive Data Register (FIFO Multiple Data, 16-bit)

**Name:** SPI_RDR (FIFO_MULTI_DATA_16)
**Offset:** 0x08
**Reset:** 0x0
**Property:** Read-only

If FIFO is enabled (FIFOEN bit in SPI_CR), refer to section Multiple Data Mode.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RD1[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RD1[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RD0[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RD0[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0:15, 16:31 – RDx** Receive Data
First unread data in the Receive FIFO. Data received by the SPI interface is stored in this register in a right-justified format. Unused bits are read as zero.

### 48.8.6 SPI Transmit Data Register

**Name:** SPI_TDR
**Offset:** 0x0C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | LASTXFER |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PCS[3:0] | | | |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TD[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TD[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 24 – LASTXFER** Last Transfer
This field is only used if variable peripheral select is active (SPI_MR.PS = 1).

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | The current NPCS is deasserted after the transfer of the character written in TD. When SPI_CSRx.CSAAT is set, the communication with the current serial peripheral can be closed by raising the corresponding NPCS line as soon as TD transfer is completed. |

**Bits 19:16 – PCS[3:0]** Peripheral Chip Select
This field is only used if variable peripheral select is active (SPI_MR.PS = 1).

If SPI_MR.PCSDEC = 0:

PCS = xxx0 NPCS[3:0] = 1110

PCS = xx01 NPCS[3:0] = 1101

PCS = x011 NPCS[3:0] = 1011

PCS = 0111 NPCS[3:0] = 0111

PCS = 1111 forbidden (no peripheral is selected)

(x = don’t care)

If SPI_MR.PCSDEC = 1:

NPCS[3:0] output signals = PCS.

**Bits 15:0 – TD[15:0]** Transmit Data
Data to be transmitted by the SPI interface is stored in this register. Information to be transmitted must be written to this register in a right-justified format.

### 48.8.7 SPI Transmit Data Register (FIFO Multiple Data, 8- to 16-bit)

**Name:** SPI_TDR (FIFO_MULTI_DATA)
**Offset:** 0x0C
**Reset:** –
**Property:** Write-only

If FIFO is enabled (FIFOEN bit in SPI_CR), refer to section Multiple Data Mode.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TD1[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TD1[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TD0[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TD0[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0:15, 16:31 – TDx** Transmit Data
Next data to write in the Transmit FIFO. Information to be transmitted must be written to this register in a right-justified format.

### 48.8.8 SPI Status Register

**Name:** SPI_SR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | | | | | | SPIENS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | | | | | CMP | UNDES | TXEMPTY | NSSR |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | OVRES | MODF | TDRE | RDRF |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 31 – RXFPTEF** Receive FIFO Pointer Error Flag
See FIFO Pointer Error for details.

| Value | Description |
|---|---|
| 0 | No Receive FIFO pointer occurred. |
| 1 | Receive FIFO pointer error occurred. Receiver must be reset. |

**Bit 30 – TXFPTEF** Transmit FIFO Pointer Error Flag
See FIFO Pointer Error for details.

| Value | Description |
|---|---|
| 0 | No Transmit FIFO pointer occurred. |
| 1 | Transmit FIFO pointer error occurred. Transceiver must be reset. |

**Bit 29 – RXFTHF** Receive FIFO Threshold Flag

| Value | Description |
|---|---|
| 0 | Number of unread data in Receive FIFO is below RXFTHRES threshold or RXFTH flag has been cleared. |
| 1 | Number of unread data in Receive FIFO has reached RXFTHRES threshold (coming from “below threshold” state to “equal or above threshold” state). |

**Bit 28 – RXFFF** Receive FIFO Full Flag

| Value | Description |
|---|---|
| 0 | Receive FIFO is not empty or RXFE flag has been cleared. |
| 1 | Receive FIFO has become full (coming from “not full” state to “full” state). |

**Bit 27 – RXFEF** Receive FIFO Empty Flag

| Value | Description |
|---|---|
| 0 | Receive FIFO is not empty or RXFE flag has been cleared. |
| 1 | Receive FIFO has become empty (coming from “not empty” state to “empty” state). |

**Bit 26 – TXFTHF** Transmit FIFO Threshold Flag (cleared on read)

| Value | Description |
|---|---|
| 0 | Number of data in Transmit FIFO is above TXFTHRES threshold. |
| 1 | Number of data in Transmit FIFO has reached TXFTHRES threshold since the last read of SPI_SR. |

**Bit 25 – TXFFF** Transmit FIFO Full Flag (cleared on read)

| Value | Description |
|---|---|
| 0 | Transmit FIFO is not full or TXFF flag has been cleared. |
| 1 | Transmit FIFO has been filled since the last read of SPI_SR. |

**Bit 24 – TXFEF** Transmit FIFO Empty Flag (cleared on read)

| Value | Description |
|---|---|
| 0 | Transmit FIFO is not empty. |
| 1 | Transmit FIFO has been emptied since the last read of SPI_SR. |

**Bit 16 – SPIENS** SPI Enable Status

| Value | Description |
|---|---|
| 0 | SPI is disabled. |
| 1 | SPI is enabled. |

**Bit 11 – CMP** Comparison Status (cleared on read)

| Value | Description |
|---|---|
| 0 | No received character matched the comparison criteria programmed in VAL1 and VAL2 fields in SPI_CMPR since the last read of SPI_SR. |
| 1 | A received character matched the comparison criteria since the last read of SPI_SR. |

**Bit 10 – UNDES** Underrun Error Status (Slave mode only) (cleared on read)

| Value | Description |
|---|---|
| 0 | No underrun has been detected since the last read of SPI_SR. |
| 1 | A transfer starts whereas no data has been loaded in SPI_TDR. |

**Bit 9 – TXEMPTY** Transmission Registers Empty (cleared by writing SPI_TDR)

| Value | Description |
|---|---|
| 0 | As soon as data is written in SPI_TDR. |
| 1 | SPI_TDR and internal shift register are empty. If a transfer delay has been defined, TXEMPTY is set after the end of this delay. |

**Bit 8 – NSSR** NSS Rising (cleared on read)

| Value | Description |
|---|---|
| 0 | No rising edge detected on NSS pin since the last read of SPI_SR. |
| 1 | A rising edge occurred on NSS pin since the last read of SPI_SR. |

**Bit 3 – OVRES** Overrun Error Status (cleared on read)
An overrun occurs when SPI_RDR is loaded at least twice from the internal shift register since the last read of SPI_RDR.

| Value | Description |
|---|---|
| 0 | No overrun has been detected since the last read of SPI_SR. |
| 1 | An overrun has occurred since the last read of SPI_SR. |

**Bit 2 – MODF** Mode Fault Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No mode fault has been detected since the last read of SPI_SR. |
| 1 | A mode fault occurred since the last read of SPI_SR. |

**Bit 1 – TDRE** Transmit Data Register Empty (cleared by writing SPI_TDR)
When FIFOs are disabled:

0: Data has been written to SPI_TDR and not yet transferred to the internal shift register.

1: The last data written in SPI_TDR has been transferred to the internal shift register.

TDRE is cleared when the SPI is disabled or at reset. Enabling the SPI sets the TDRE flag.

When FIFOs are enabled:

0: Transmit FIFO is full and cannot accept more data.

1: Transmit FIFO is not full; one or more data can be written according to TXRDYM field configuration.

TDRE behavior with FIFOs enabled is illustrated in section TXEMPTY, TDRE and RDRF Behavior.

**Bit 0 – RDRF** Receive Data Register Full (cleared by reading SPI_RDR)
When FIFOs are disabled:

0: No data has been received since the last read of SPI_RDR.

1: Data has been received and the received data has been transferred from the internal shift register to SPI_RDR since the last read of SPI_RDR.

When FIFOs are enabled:

0: Receive FIFO is empty; no data to read.

1: At least one unread data is in the Receive FIFO.

RDRF behavior with FIFOs enabled is illustrated in section TXEMPTY, TDRE and RDRF Behavior.

### 48.8.9 SPI Interrupt Enable Register

**Name:** SPI_IER
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CMP | UNDES | TXEMPTY | NSSR |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | MODF | TDRE | RDRF |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

**Bit 31 – RXFPTEF** RXFPTEF Interrupt Enable

**Bit 30 – TXFPTEF** TXFPTEF Interrupt Enable

**Bit 29 – RXFTHF** RXFTHF Interrupt Enable

**Bit 28 – RXFFF** RXFFF Interrupt Enable

**Bit 27 – RXFEF** RXFEF Interrupt Enable

**Bit 26 – TXFTHF** TXFTHF Interrupt Enable

**Bit 25 – TXFFF** TXFFF Interrupt Enable

**Bit 24 – TXFEF** TXFEF Interrupt Enable

**Bit 11 – CMP** Comparison Interrupt Enable

**Bit 10 – UNDES** Underrun Error Interrupt Enable

**Bit 9 – TXEMPTY** Transmission Registers Empty Enable

**Bit 8 – NSSR** NSS Rising Interrupt Enable

**Bit 3 – OVRES** Overrun Error Interrupt Enable

**Bit 2 – MODF** Mode Fault Error Interrupt Enable

**Bit 1 – TDRE** SPI Transmit Data Register Empty Interrupt Enable

**Bit 0 – RDRF** Receive Data Register Full Interrupt Enable

© 2018 Microchip Technology Inc.

### 48.8.10 SPI Interrupt Disable Register

**Name:** SPI_IDR
**Offset:** 0x18
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CMP | UNDES | TXEMPTY | NSSR |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | MODF | TDRE | RDRF |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

**Bit 31 – RXFPTEF** RXFPTEF Interrupt Disable

**Bit 30 – TXFPTEF** TXFPTEF Interrupt Disable

**Bit 29 – RXFTHF** RXFTHF Interrupt Disable

**Bit 28 – RXFFF** RXFFF Interrupt Disable

**Bit 27 – RXFEF** RXFEF Interrupt Disable

**Bit 26 – TXFTHF** TXFTHF Interrupt Disable

**Bit 25 – TXFFF** TXFFF Interrupt Disable

**Bit 24 – TXFEF** TXFEF Interrupt Disable

**Bit 11 – CMP** Comparison Interrupt Disable

**Bit 10 – UNDES** Underrun Error Interrupt Disable

**Bit 9 – TXEMPTY** Transmission Registers Empty Disable

**Bit 8 – NSSR** NSS Rising Interrupt Disable

**Bit 3 – OVRES** Overrun Error Interrupt Disable

**Bit 2 – MODF** Mode Fault Error Interrupt Disable

**Bit 1 – TDRE** SPI Transmit Data Register Empty Interrupt Disable

**Bit 0 – RDRF** Receive Data Register Full Interrupt Disable

© 2018 Microchip Technology Inc.

### 48.8.11 SPI Interrupt Mask Register

**Name:** SPI_IMR
**Offset:** 0x1C
**Reset:** 0x0
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RXFPTEF | TXFPTEF | RXFTHF | RXFFF | RXFEF | TXFTHF | TXFFF | TXFEF |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CMP | UNDES | TXEMPTY | NSSR |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | MODF | TDRE | RDRF |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 31 – RXFPTEF** RXFPTEF Interrupt Mask

**Bit 30 – TXFPTEF** TXFPTEF Interrupt Mask

**Bit 29 – RXFTHF** RXFTHF Interrupt Mask

**Bit 28 – RXFFF** RXFFF Interrupt Mask

**Bit 27 – RXFEF** RXFEF Interrupt Mask

**Bit 26 – TXFTHF** TXFTHF Interrupt Mask

**Bit 25 – TXFFF** TXFFF Interrupt Mask

**Bit 24 – TXFEF** TXFEF Interrupt Mask

**Bit 11 – CMP** Comparison Interrupt Mask

**Bit 10 – UNDES** Underrun Error Interrupt Mask

**Bit 9 – TXEMPTY** Transmission Registers Empty Mask

**Bit 8 – NSSR** NSS Rising Interrupt Mask

**Bit 3 – OVRES** Overrun Error Interrupt Mask

**Bit 2 – MODF** Mode Fault Error Interrupt Mask

**Bit 1 – TDRE** SPI Transmit Data Register Empty Interrupt Mask

**Bit 0 – RDRF** Receive Data Register Full Interrupt Mask

### 48.8.12 SPI Chip Select Register

**Name:** SPI_CSRx
**Offset:** 0x30 + x*0x04 [x=0..3]
**Reset:** 0
**Property:** R/W

This register can only be written if the WPEN bit is cleared in the SPI Write Protection Mode Register.

SPI_CSRx must be written even if the user wants to use the default reset values. The BITS field is not updated with the translated value unless the register is written.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DLYBCT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DLYBS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SCBR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BITS[3:0] | | | | CSAAT | CSNAAT | NCPHA | CPOL |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – DLYBCT[7:0]** Delay Between Consecutive Transfers
This field defines the delay between two consecutive transfers with the same peripheral without removing the chip select. The delay is always inserted after each transfer and before removing the chip select if needed.

When DLYBCT = 0, no delay between consecutive transfers is inserted and the clock keeps its duty cycle over the character transfers.

Otherwise, the following equations determine the delay:

If SPI_MR.BRSRCCLK = 0: DLYBCT = Delay Between Consecutive Transfers × $f_{peripheral\ clock}$ / 32

If SPI_MR.BRSRCCLK = 1: DLYBCT = Delay Between Consecutive Transfers × $f_{GCLK}$ / 32

**Bits 23:16 – DLYBS[7:0]** Delay Before SPCK
This field defines the delay from NPCS falling edge (activation) to the first valid SPCK transition.

When DLYBS = 0, the delay is half the SPCK clock period.

Otherwise, the following equations determine the delay:

If SPI_MR.BRSRCCLK = 0: DLYBS = Delay Before SPCK × $f_{peripheral\ clock}$

If SPI_MR.BRSRCCLK = 1: DLYBS = Delay Before SPCK × $f_{GCLK}$

**Bits 15:8 – SCBR[7:0]** Serial Clock Bit Rate
In Master mode, the SPI Interface uses a modulus counter to derive the SPCK bit rate from the clock defined by SPI_MR.BRSRCCLK bit. The bit rate is selected by writing a value from1 to 255 in the SCBR field. The following equations determine the SPCK bit rate:

If SPI_MR.BRSRCCLK = 0: SCBR = $f_{peripheral\ clock}$ / SPCK Bit Rate

If SPI_MR.BRSRCCLK = 1: SCBR = $f_{GCLK}$ / SPCK Bit Rate

Programming the SCBR field to 0 is forbidden. Triggering a transfer while SCBR is at 0 can lead to unpredictable results.

If SPI_MR.BRSRCCLK = 1, SCBR must be programmed with a value greater than 1.

At reset, SCBR is 0 and the user has to program it at a valid value before performing the first transfer.

**Note:** If one of the SCBR fields in SPI_CSRx is set to 1, the other SCBR fields in SPI_CSRx must be set to 1 as well, if they are used to process transfers. If they are not used to transfer data, they can be set at any value.

**Bits 7:4 – BITS[3:0]** Bits Per Transfer
(See Note under the register table in SPI Chip Select Register.)

The BITS field determines the number of data bits transferred. Reserved values should not be used.

| Value | Name | Description |
|---|---|---|
| 0 | 8_BIT | 8 bits for transfer |
| 1 | 9_BIT | 9 bits for transfer |
| 2 | 10_BIT | 10 bits for transfer |
| 3 | 11_BIT | 11 bits for transfer |
| 4 | 12_BIT | 12 bits for transfer |
| 5 | 13_BIT | 13 bits for transfer |
| 6 | 14_BIT | 14 bits for transfer |
| 7 | 15_BIT | 15 bits for transfer |
| 8 | 16_BIT | 16 bits for transfer |
| 9 | – | Reserved |
| 10 | – | Reserved |
| 11 | – | Reserved |
| 12 | – | Reserved |
| 13 | – | Reserved |
| 14 | – | Reserved |
| 15 | – | Reserved |

**Bit 3 – CSAAT** Chip Select Active After Transfer

| Value | Description |
|---|---|
| 0 | The Peripheral Chip Select Line rises as soon as the last transfer is achieved. |
| 1 | The Peripheral Chip Select Line does not rise after the last transfer is achieved. It remains active until a new transfer is requested on a different chip select. |

**Bit 2 – CSNAAT** Chip Select Not Active After Transfer (ignored if CSAAT = 1)

| Value | Description |
|---|---|
| 0 | The Peripheral Chip Select Line does not rise between two transfers if SPI_TDR is reloaded before the end of the first transfer and if the two transfers occur on the same chip select. |
| 1 | The Peripheral Chip Select Line rises systematically after each transfer performed on the same slave. It remains inactive after the end of transfer for a minimal duration of:<br>If SPI_MR.BRSRCCLK = 0:<br>$\frac{\mathrm{DLYBCS}}{f_{\text{peripheral clock}}}$<br>If SPI_MR.BRSRCCLK = 1:<br>$\frac{\mathrm{DLYBCS}}{f_{\mathrm{GCLK}}}$<br>If field DLYBCS is lower than 6, a minimum of six periods is introduced. |

**Bit 1 – NCPHA** Clock Phase
NCPHA determines which edge of SPCK causes data to change and which edge causes data to be captured. NCPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.

| Value | Description |
|---|---|
| 0 | Data is changed on the leading edge of SPCK and captured on the following edge of SPCK. |
| 1 | Data is captured on the leading edge of SPCK and changed on the following edge of SPCK. |

**Bit 0 – CPOL** Clock Polarity
CPOL is used to determine the inactive state value of the serial clock (SPCK). It is used with NCPHA to produce the required clock/data relationship between master and slave devices.

| Value | Description |
|---|---|
| 0 | The inactive state value of SPCK is logic level zero. |
| 1 | The inactive state value of SPCK is logic level one. |

### 48.8.13 SPI FIFO Mode Register

**Name:** SPI_FMR
**Offset:** 0x40
**Reset:** 0x0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | RXFTHRES[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | TXFTHRES[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | RXRDYM[1:0] | | | | TXRDYM[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bits 29:24 – RXFTHRES[5:0]** Receive FIFO Threshold

| Value | Description |
|---|---|
| 0–16 | Defines the Receive FIFO threshold value (number of data). SPI_SR.RXFTH will be set when the Receive FIFO goes from “below” threshold state to “equal or above” threshold state. |

**Bits 21:16 – TXFTHRES[5:0]** Transmit FIFO Threshold

| Value | Description |
|---|---|
| 0–16 | Defines the Transmit FIFO threshold value (number of data). SPI_SR.TXFTH will be set when the Transmit FIFO goes from “above” threshold state to “equal or below” threshold state. |

**Bits 5:4 – RXRDYM[1:0]** Receive Data Register Full Mode
If FIFOs are enabled, the SPI_SR.RDRF flag behaves as follows:

| Value | Name | Description |
|---|---|---|
| 0 | ONE_DATA | RDRF will be at level ‘1’ when at least one unread data is in the Receive FIFO. |

| Value | Name | Description |
|---|---|---|
| 1 | TWO_DATA | RDRF will be at level ‘1’ when at least two unread data are in the Receive FIFO.<br>Cannot be used when SPI_MR.MSTR =1, or if SPI_MR.PS =1. |
| 2 | FOUR_DATA | RDRF will be at level ‘1’ when at least four unread data are in the Receive FIFO.<br>Cannot be used when SPI_CSRx.BITS is greater than 0, or if SPI_MR.MSTR =1, or if SPI_MR.PS =1. |

**Bits 1:0 – TXRDYM[1:0]** Transmit Data Register Empty Mode
If FIFOs are enabled, the SPI_SR.TDRE flag behaves as follows:

| Value | Name | Description |
|---|---|---|
| 0 | ONE_DATA | TDRE will be at level ‘1’ when at least one data can be written in the Transmit FIFO. |
| 1 | TWO_DATA | TDRE will be at level ‘1’ when at least two data can be written in the Transmit FIFO.<br>Cannot be used if SPI_MR.PS =1. |

### 48.8.14 SPI FIFO Level Register

**Name:** SPI_FLR
**Offset:** 0x44
**Reset:** 0x0
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | RXFL[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TXFL[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 21:16 – RXFL[5:0]** Receive FIFO Level

| Value | Description |
|---|---|
| 0 | There is no unread data in the Receive FIFO. |
| 1-16 | Indicates the number of unread data in the Receive FIFO. |

**Bits 5:0 – TXFL[5:0]** Transmit FIFO Level

| Value | Description |
|---|---|
| 0 | There is no data in the Transmit FIFO. |
| 1-16 | Indicates the number of data in the Transmit FIFO. |

### 48.8.15 SPI Comparison Register

**Name:** SPI_CMPR
**Offset:** 0x48
**Reset:** 0x0
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the SPI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | VAL2[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | VAL2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | VAL1[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | VAL1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – VAL2[15:0]** Second Comparison Value for Received Character

| Value | Description |
|---|---|
| 0-65535 | The received character must be lower or equal to the value of VAL2 and higher or equal to VAL1 to set CMP flag in SPI_CSR. If asynchronous partial wakeup (SleepWalking) is enabled in PMC_SLPWK_ER, the SPI requests a system wakeup if condition is met. |

**Bits 15:0 – VAL1[15:0]** First Comparison Value for Received Character

| Value | Description |
|---|---|
| 0-65535 | The received character must be higher or equal to the value of VAL1 and lower or equal to VAL2 to set CMP flag in SPI_SR. If asynchronous partial wakeup (SleepWalking) is enabled in PMC_SLPWK_ER, the SPI requests a system wakeup if the condition is met. |

### 48.8.16 SPI Write Protection Mode Register

**Name:** SPI_WPMR
**Offset:** 0xE4
**Reset:** 0x0
**Property:** Read/Write

See section Register Write Protection for the list of registers that can be write-protected.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | WPKEY[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | WPKEY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x535049 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit.<br>Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x535049 (“SPI” in ASCII) |
| 1 | Enables the write protection if WPKEY corresponds to 0x535049 (“SPI” in ASCII) |

### 48.8.17 SPI Write Protection Status Register

**Name:** SPI_WPSR
**Offset:** 0xE8
**Reset:** 0x0
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 15:8 – WPVSRC[7:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of SPI_WPSR. |
| 1 | A write protection violation has occurred since the last read of SPI_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 49. Quad Serial Peripheral Interface (QSPI)

## 49.1 Description

The Quad Serial Peripheral Interface (QSPI) is a synchronous serial data link that provides communication with external devices in Master mode.

The QSPI can be used in SPI mode to interface to serial peripherals such as ADCs, DACs, LCD controllers, CAN controllers and sensors, or in Serial Memory mode to interface to serial Flash memories.

The QSPI allows the system to execute code directly from a serial Flash memory (XIP) without code shadowing to RAM. The serial Flash memory mapping is seen in the system as other memories such as ROM, SRAM, DRAM, embedded Flash memory, etc.

With the support of the Quad SPI protocol, the QSPI allows the system to use high-performance serial Flash memories which are small and inexpensive, in place of larger and more expensive parallel Flash memories.

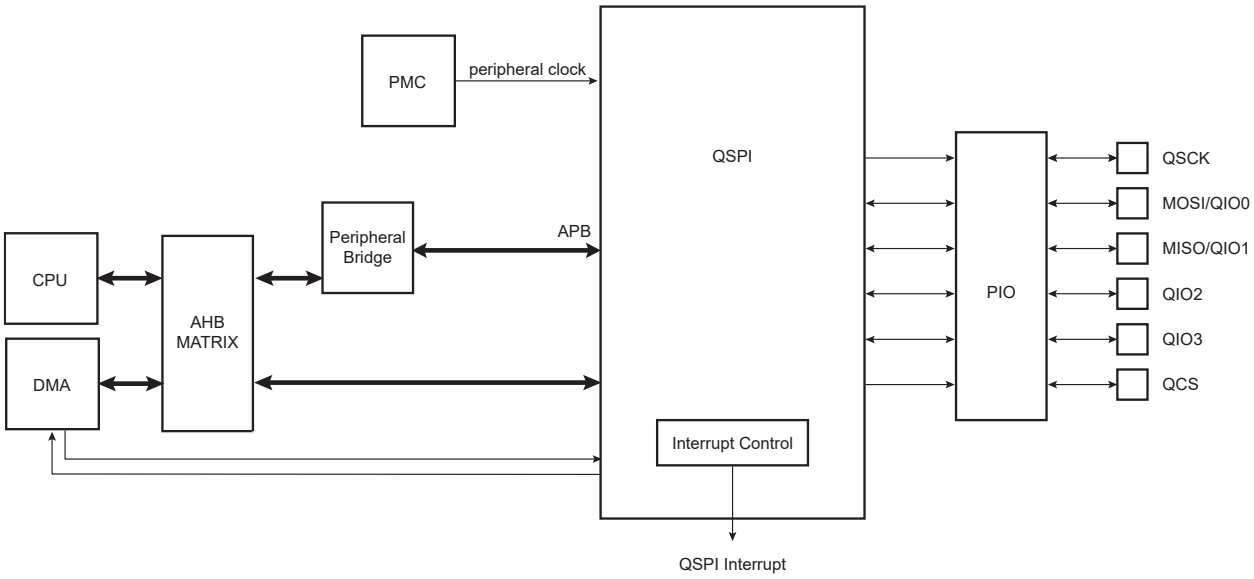
**Note:** Stacked devices with a rollover in the memory address space at each die boundary are not supported.

## 49.2 Embedded Characteristics

- Master SPI Interface
  - Programmable clock phase and clock polarity
  - Programmable transfer delays between consecutive transfers, between clock and data, between deactivation and activation of chip select
- SPI Mode
  - Interface to serial peripherals such as ADCs, DACs, LCD controllers, CAN controllers and sensors
  - 8-bit/16-bit/32-bit programmable data length
- Serial Memory Mode
  - Interface to serial Flash memories operating in Single-bit SPI, Dual SPI and Quad SPI
  - Interface to serial Flash Memories operating in Single Data Rate Mode
  - Supports “Execute In Place” (XIP)— code execution by the system directly from a serial Flash memory
  - Flexible instruction register for compatibility with all serial Flash memories
  - 32-bit address mode (default is 24-bit address) to support serial Flash memories larger than 128 Mbits
  - Continuous read mode
  - Scrambling/unscrambling “On-The-Fly”
- Connection to DMA Channel Capabilities Optimizes Data Transfers
  - One channel for the receiver, one channel for the transmitter
- Register Write Protection

## 49.3 Block Diagram

**Figure 49-1. Block Diagram**




## 49.4 Signal Description

**Table 49-1. Signal Description**

| Pin Name | Pin Description | Type |
|---|---|---|
| QSCK | Serial Clock | Output |
| MOSI (QIO0) (1)(2) | Data Output (Data Input Output 0) | Output (Input/Output) |
| MISO (QIO1) (1)(2) | Data Input (Data Input Output 1) | Input (Input/Output) |
| QIO2 (3) | Data Input Output 2 | Input/Output |
| QIO3 (3) | Data Input Output 3 | Input/Output |
| QCS | Peripheral Chip Select | Output |

**Note:**

1. MOSI and MISO are used for single-bit SPI operation.
2. QIO0–QIO1 are used for Dual SPI operation.
3. QIO0–QIO3 are used for Quad SPI operation.

## 49.5 Product Dependencies

### 49.5.1 I/O Lines

The pins used for interfacing the compliant external devices may be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the QSPI pins to their peripheral functions.

### 49.5.2 Power Management

The QSPI may be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the QSPI clock.

### 49.5.3 Interrupt Sources

The QSPI has an interrupt line connected to the Interrupt Controller. Handling the QSPI interrupt requires programming the interrupt controller before configuring the QSPI.

### 49.5.4 Direct Memory Access Controller (DMA)

The QSPI can be used in conjunction with the Direct Memory Access Controller (DMA) in order to reduce processor overhead. For a full description of the DMA, refer to the section “DMA Controller (XDMAC)”.

## 49.6 Functional Description

### 49.6.1 Serial Clock Baud Rate

The QSPI baud rate clock is generated by dividing the peripheral clock by a value between 1 and 256.

### 49.6.2 Serial Clock Phase and Polarity

Four combinations of polarity and phase are available for data transfers. The clock polarity is programmed with the CPOL bit in the QSPI Serial Clock register (QSPI_SCR). The CPHA bit in the QSPI_SCR programs the clock phase. These two parameters determine the edges of the clock signal on which data is driven and sampled. Each of the two parameters has two possible states, resulting in four possible combinations that are incompatible with one another. Thus, the interfaced slave must use the same parameter values to communicate.

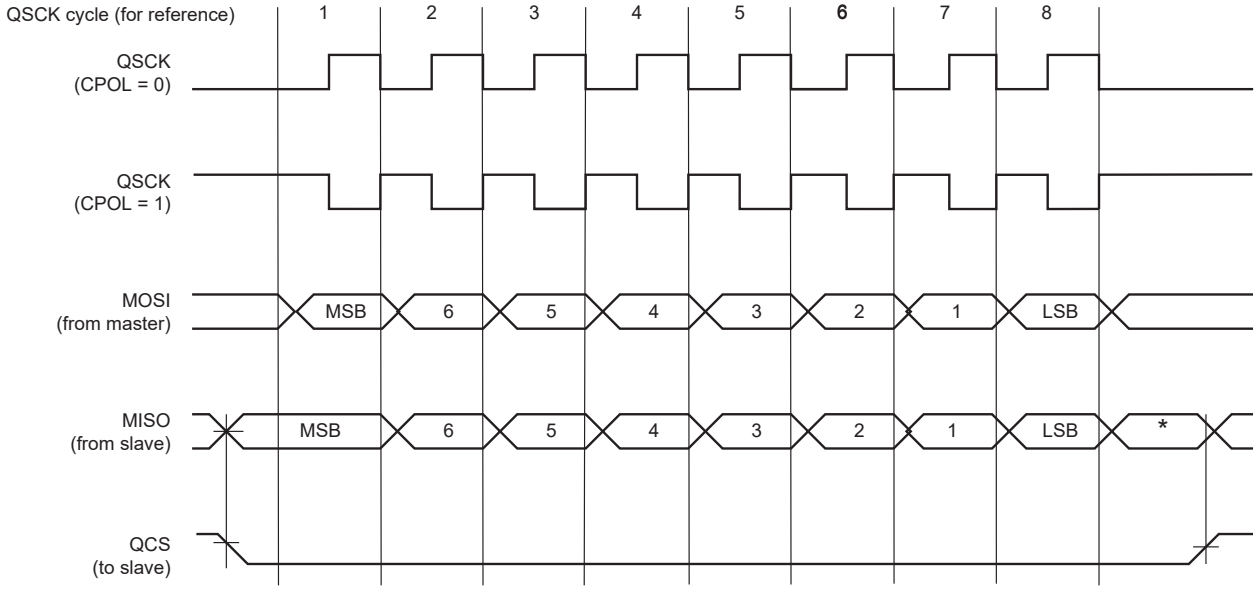
The table below shows the four modes and corresponding parameter settings.

**Table 49-2. QSPI Bus Clock Modes**

| QSPI Clock Mode | QSPI_SCR.CPOL | QSPI_SCR.CPHA | Shift QSCK Edge | Capture QSCK Edge | QSCK Inactive Level |
|---|---|---|---|---|---|
| 0 | 0 | 0 | Falling | Falling | Low |
| 1 | 0 | 1 | Rising | Rising | Low |
| 2 | 1 | 0 | Rising | Rising | High |
| 3 | 1 | 1 | Falling | Falling | High |

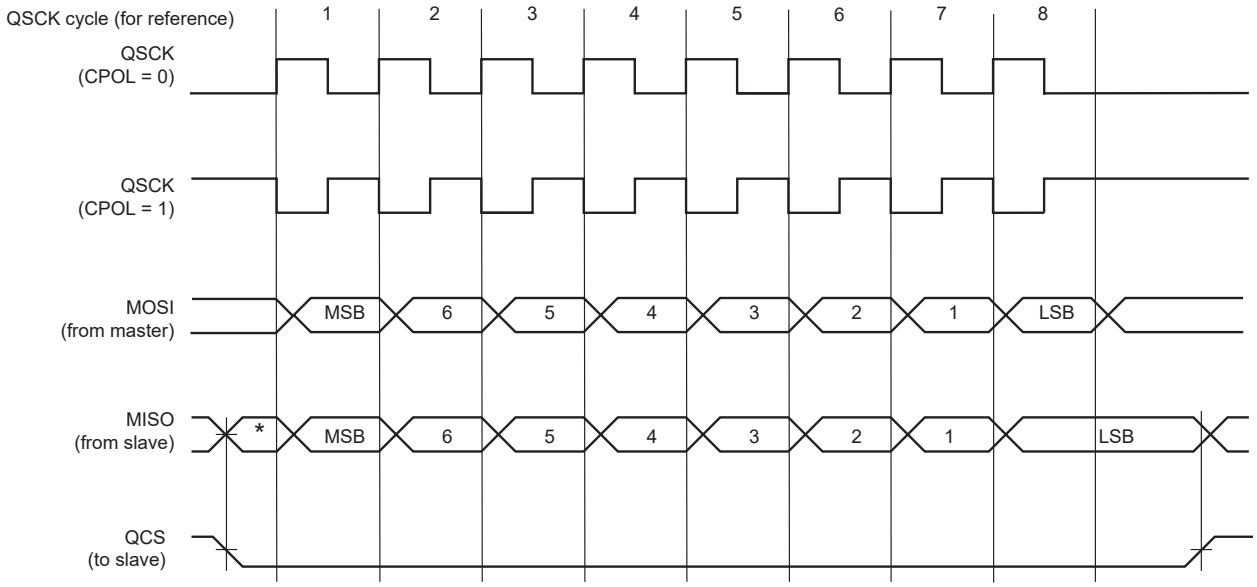
The following figures show examples of data transfers.

**Figure 49-2. QSPI Transfer Format (QSPI_SCR.CPHA = 0, 8 bits per transfer)**




* Not defined, but normally MSB of previous character received.

**Figure 49-3. QSPI Transfer Format (QSPI_SCR.CPHA = 1, 8 bits per transfer)**




* Not defined but normally LSB of previous character transmitted.

### 49.6.3 Transfer Delays

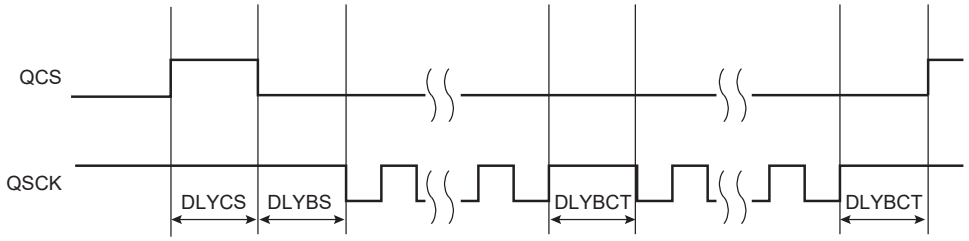
The figure below shows several consecutive transfers while the chip select is active. Three delays can be programmed to modify the transfer waveforms:

- The delay between the deactivation and the activation of QCS, programmed by writing QSPI_MR.DLYCS. Allows to adjust the minimum time of QCS at high level.
- The delay before QSCK, programmed by writing QSPI_SR.DLYBS. Allows the start of QSCK to be delayed after the chip select has been asserted.

- The delay between consecutive transfers, programmed by writing QSPI_MR.DLYBCT. Allows insertion of a delay between two consecutive transfers. In Serial Memory mode, this delay is not programmable and DLYBCT is ignored. In this mode, DLYBCT must be written to ‘0’.

These delays allow the QSPI to be adapted to the interfaced peripherals and their speed and bus release time.

**Figure 49-4. Programmable Delays**




### 49.6.4 QSPI SPI Mode

In SPI mode, the QSPI acts as a standard SPI Master.

To activate this mode, QSPI_MR.SMM must be written to ‘0’ in QSPI_MR.

#### 49.6.4.1 SPI Mode Operations

The QSPI in standard SPI mode operates on the clock generated by the internal programmable baud rate generator. It fully controls the data transfers to and from the slave connected to the SPI bus. The QSPI drives the chip select line to the slave (QCS) and the serial clock signal (QSCK).

The QSPI features two holding registers, the Transmit Data register (QSPI_TDR) and the Receive Data register (QSPI_RDR), and a single internal shift register. The holding registers maintain the data flow at a constant rate.

After enabling the QSPI, a data transfer begins when the processor writes to the QSPI_TDR. The written data is immediately transferred to the internal shift register and transfer on the SPI bus starts. While the data in the internal shift register is shifted on the MOSI line, the MISO line is sampled and shifted to the internal shift register. Receiving data cannot occur without transmitting data. If receiving mode is not needed, for example when communicating with a slave receiver only (such as an LCD), the receive status flags in the Status register (QSPI_SR) can be discarded.

If new data is written in QSPI_TDR during the transfer, it is retained there until the current transfer is completed. Then, the received data is transferred from the internal shift register to the QSPI_RDR, the data in QSPI_TDR is loaded in the internal shift register and a new transfer starts.

The transfer of a data written in QSPI_TDR in the internal shift register is indicated by the Transmit Data Register Empty (TDRE) bit in the QSPI_SR. When new data is written in QSPI_TDR, this bit is cleared. QSPI_SR.TDRE is used to trigger the Transmit DMA channel.

The end of transfer is indicated by the TXEMPTY flag in the QSPI_SR. If a transfer delay (DLYBCT) is greater than 0 for the last transfer, QSPI_SR.TXEMPTY is set after the completion of this delay. The peripheral clock can be switched off at this time.

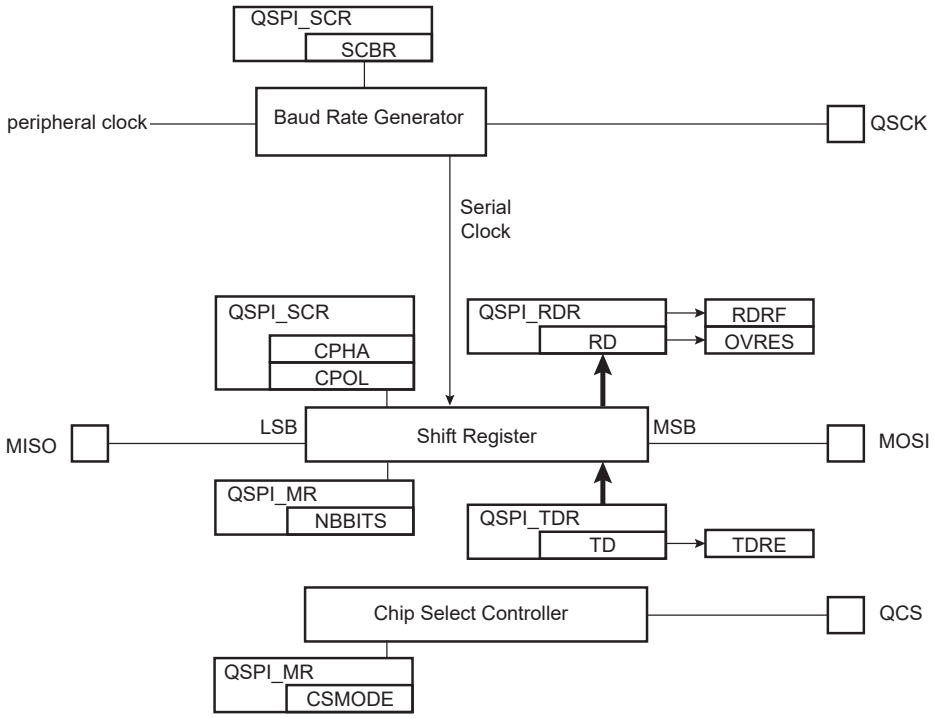
The transfer of received data from the internal shift register in QSPI_RDR is indicated by the Receive Data Register Full (RDRF) bit in the QSPI_SR. When the received data is read, QSPI_SR.RDRF bit is cleared.

If the QSPI_RDR has not been read before new data is received, the Overrun Error Status (OVRES) bit in QSPI_SR is set. As long as this flag is set, data is loaded in QSPI_RDR. The user must read the QSPI_SR to clear the OVRES bit.

The following figures show, respectively, a block diagram of the SPI when operating in Master mode, and a flow chart describing how transfers are handled.

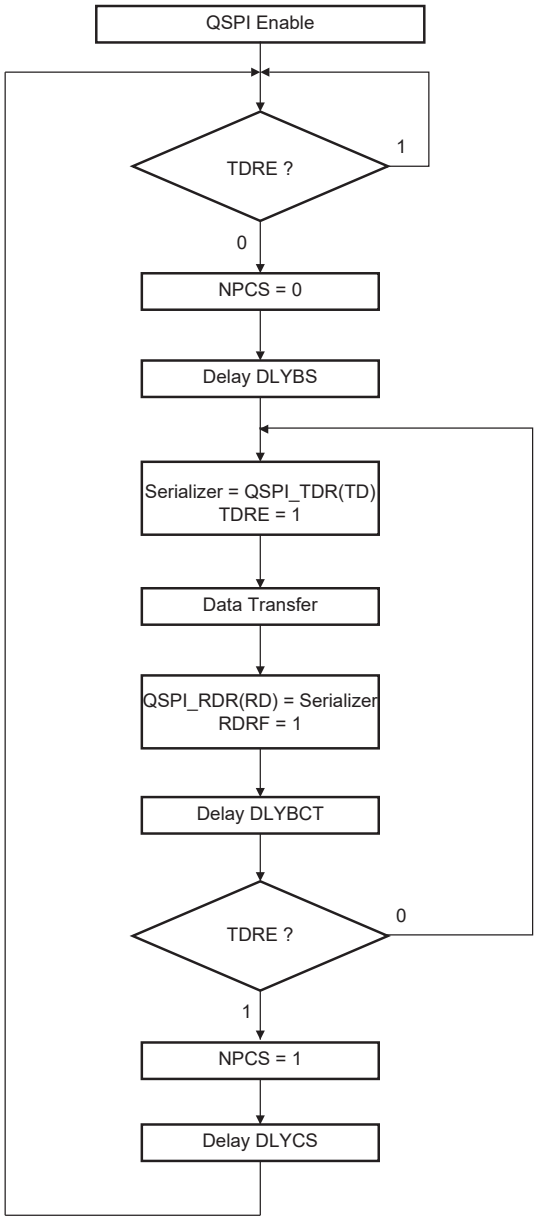
#### 49.6.4.2 SPI Mode Block Diagram

**Figure 49-5. SPI Mode Block Diagram**



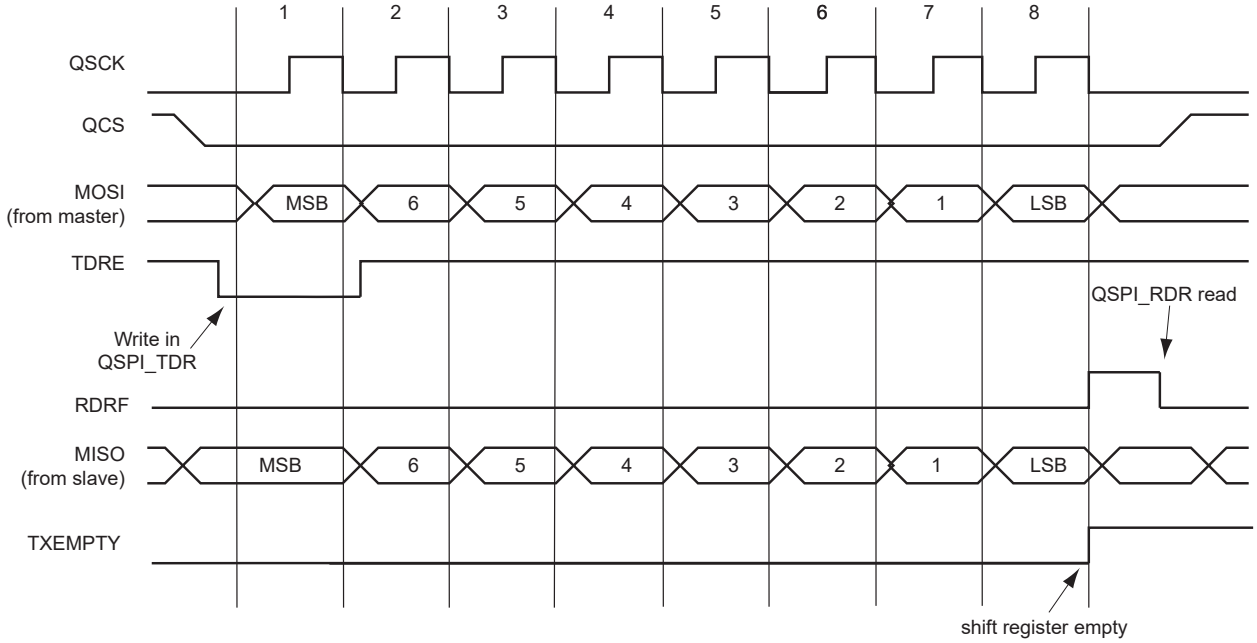
#### 49.6.4.3 SPI Mode Flow Diagram

**Figure 49-6. SPI Mode Flow Diagram**




The figure below shows Transmit Data Register Empty (TDRE), Receive Data Register Full (RDRF) and Transmission Register Empty (TXEMPTY) status flags behavior within the QSPI_SR during an 8-bit data transfer in Fixed mode, without DMA.

**Figure 49-7. Status Register Flags Behavior**




#### 49.6.4.4 Peripheral Deselection without DMA

During a transfer of more than one data on a Chip Select without the DMA, the QSPI_TDR is loaded by the processor and the flag TDRE rises as soon as the content of the QSPI_TDR is transferred into the internal shift register. When this flag is detected high, the QSPI_TDR can be reloaded. If this reload by the processor occurs before the end of the current transfer, the Chip Select is not deasserted between the two transfers. Depending on the application software handling the QSPI_SR flags (by interrupt or polling method) or servicing other interrupts or other tasks, the processor may not reload the QSPI_TDR in time to keep the chip select active (low). A null Delay Between Consecutive Transfer (DLYBCT) value in the QSPI_MR gives even less time for the processor to reload the QSPI_TDR. With some SPI slave peripherals, requiring the chip select line to remain active (low) during a full set of transfers may lead to communication errors.

To facilitate interfacing with such devices, QSPI_MR.CSMODE may be configured to ‘1’. This allows the chip select lines to remain in their current state (low = active) until the end of transfer is indicated by the Last Transfer (LASTXFER) bit in the Control register (QSPI_CR). Even if the QSPI_TDR is not reloaded, the chip select remains active. To have the chip select line rise at the end of the last data transfer, QSPI_CR.LASTXFER must be written to ‘1’ at the same time or after writing the last data to transmit into the QSPI_TDR.

#### 49.6.4.5 Peripheral Deselection with DMA

When the DMA Controller is used, the Chip Select line remains low during the transfer since the TDRE flag is managed by the DMA itself. Reloading the QSPI_TDR by the DMA is done as soon as the TDRE flag is set. In this case, writing QSPI_MR.CSMODE to ‘1’ may not be needed. However, when other DMA channels connected to other peripherals are also in use, the QSPI DMA could be delayed by another DMA with a higher priority on the bus. Having DMA buffers in slower memories like Flash memory or SDRAM compared to fast internal SRAM, may lengthen the reload time of the QSPI_TDR by the DMA as well. This means that the QSPI_TDR might not be reloaded in time to keep the chip select line low. In this case, the chip select line may toggle between data transfer and according to some SPI Slave devices, the communication might get lost. It may be necessary to configure QSPI_MR.CSMODE to ‘1’.

When QSPI_MR.CSMODE is configured to ‘0’, the QCS does not rise in all cases between two transfers on the same peripheral. During a transfer on a Chip Select, the flag TDRE rises as soon as the content of the QSPI_TDR is transferred into the internal shifter. When this flag is detected, the QSPI_TDR can be reloaded. If this reload occurs before the end of the current transfer, the Chip Select is not deasserted between the two transfers. This might lead to difficulties for interfacing with some serial peripherals requiring the chip select to be deasserted after each transfer. To facilitate interfacing with such devices, the QSPI_MR may be configured with QSPI_MR.CSMODE at ‘2’.

### 49.6.5 QSPI Serial Memory Mode

In Serial Memory mode, the QSPI acts as a serial Flash memory controller. The QSPI can be used to read data from the serial Flash memory allowing the CPU to execute code from it (XIP execute in place). The QSPI can also be used to control the serial Flash memory (Program, Erase, Lock, etc.) by sending specific commands. In this mode, the QSPI is compatible with single-bit SPI, Dual SPI and Quad SPI protocols.

To activate this mode, QSPI_MR.SMM must be written to ‘1’.

In Serial Memory mode, data is transferred either by QSPI_TDR and QSPI_RDR or by writing or read in the QSPI memory space (0x90000000/0x98000000/0xD0000000/0xD8000000) depending on TFRTYP and SMRM configuration.

#### 49.6.5.1 Instruction Frame

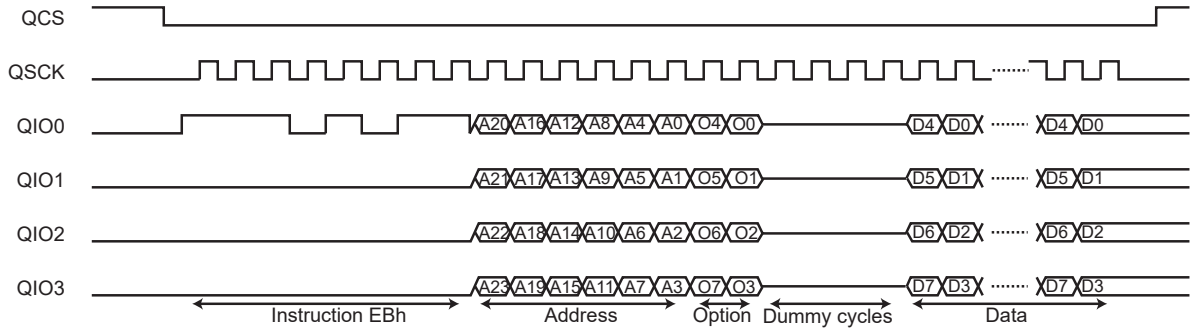
In order to control serial Flash memories, the QSPI is able to send instructions via the SPI bus (ex: READ, PROGRAM, ERASE, LOCK, etc.). Because the instruction set implemented in serial Flash memories is memory vendor-dependent, the QSPI includes a complete Instruction Frame register (QSPI_IFR), which makes it very flexible and compatible with all serial Flash memories.

An instruction frame includes:

- An instruction code (size: 8 bits). The instruction is optional in some cases (see section Continuous Read mode).
- An address (size: 24 bits or 32 bits). The address is optional but is required by instructions such as READ, PROGRAM, ERASE, LOCK. By default the address is 24 bits long, but it can be 32 bits long to support serial Flash memories larger than 128 Mbits (16 Mbytes).
- An option code (size: 1/2/4/8 bits). The option code is not required, but it is useful to activate the XIP mode or the Continuous Read mode (see section Continuous Read mode) for READ instructions, in some serial Flash memory devices. These modes improve the data read latency.
- Dummy cycles. Dummy cycles are optional but required by some READ instructions.
- Data bytes are optional. Data bytes are present for data transfer instructions such as READ or PROGRAM.

The instruction code, the address/option and the data can be sent with Single-bit SPI, Dual SPI or Quad SPI protocols.

**Figure 49-8. Instruction Frame**




### 49.6.5.2 Instruction Frame Transmission

To send an instruction frame, the user must first configure the address to send by writing the field ADDR in the Instruction Address register (QSPI_IAR). This step is required if the instruction frame includes an address and no data. When data is present, the address of the instruction is defined by the address of the data accesses in the QSPI memory space, not by QSPI_IAR.

If the instruction frame includes the instruction code and/or the option code, the user must configure the instruction code and/or the option code to send by writing the fields INST and OPT in the Instruction Code register (QSPI_ICR).

Then, the user must write QSPI_IFR to configure the instruction frame depending on which instruction must be sent. If the instruction frame does not include data, writing in this register triggers the send of the instruction frame in the QSPI. If the instruction frame includes data, the send of the instruction frame is triggered by the first data access in the QSPI memory space.

The instruction frame is configured by the following bits and fields of QSPI_IFR:

- WIDTH field—used to configure which data lanes are used to send the instruction code, the address, the option code and to transfer the data. It is possible to use two unidirectional data lanes (MISO-MOSI Single-bit SPI), two bidirectional data lanes (QIO0-QIO1 Dual SPI) or four bidirectional data lanes (QIO0–QIO3 Quad SPI).
- INSTEN bit—used to enable the send of an instruction code.
- ADDREN bit—used to enable the send of an address after the instruction code.
- OPTEN bit—used to enable the send of an option code after the address.
- DATAEN bit—used to enable the transfer of data (READ or PROGRAM instruction).
- OPTL field—used to configure the option code length. The value written in OPTL must be consistent with the value written in the field WIDTH. For example: OPTL = 0 (1-bit option code) is not consistent with WIDTH = 6 (option code sent with QuadSPI protocol, thus the minimum length of the option code is 4 bits).
- ADDRL bit—used to configure the address length.
- TFRTYP field—used to define which type of data transfer must be performed.
- NBDUM field—used to configure the number of dummy cycles when reading data from the serial Flash memory. Between the address/option and the data, with some instructions, dummy cycles are inserted by the serial Flash memory.

Refer to 49.6.5.2 Instruction Frame Transmission.

If data transfer is enabled, the user can access the serial memory by reading or writing the QSPI memory space:

- To read in the serial memory, but not a memory data, for example a JEDEC-ID or the QSPI_SR, QSPI_IFR.TFRTYP must be written to ‘0’.
- To read in the serial memory, and particularly a memory data, TFRTYP must be written to ‘1’.
- To write in the serial memory, but not a memory data, for example writing the configuration or the QSPI_SR, TFRTYP must be written to ‘2’.
- If the user wants to write in the serial memory in particular to program a memory data, TFRTYP must be written to ‘3’ .

If QSPI_IFR.TFRTYP has a value other than ‘1’ and QSPI_MR.SMRM = 0, the address sent in the instruction frame is the address of the first system bus accesses. The addresses of the next accesses are not used by the QSPI. At each system bus access, an SPI transfer is performed with the same size. For example, a halfword system bus access leads to a 16-bit SPI transfer, and a byte system bus access leads to an 8-bit SPI transfer.

If SMRM = 1 and TFRTYP= (0 or 2), accesses are made via the QSPI registers and the address sent in the instruction frame is the address defined in QSPI_IAR.

Each time QSPI_IFR is written (in case of read access), or each time QSPI_TDR is written (in case of write transfer), an SPI transfer is performed with a byte size. Another byte is read each time QSPI_RDR is read (flag RDRF shows when a data can be read in QSPI_RDR) or written each time QSPI_TDR is written (flag TDRE shows when a new data can be written). The SPI transfer ends by writing QSPI_CR.LASTXFER.

If TFRTYP = 1, the address of the first instruction frame is the one of the first read access in the QSPI memory space. Each time the read accesses become nonsequential (addresses are not consecutive), a new instruction frame is sent with the last system bus access address. In this way, the system can read data at a random location in the serial memory. The size of the SPI transfers may differ from the size of the system bus read accesses.

When data transfer is not enabled, the end of the instruction frame is indicated when QSPI_SR.INSTRE rises. (The QSPI_SR.CSR flag indicates when chip select rises. A delay between these flags may exist in case of high clock division or a high DLYBCT value).

When data transfer is enabled, the user must indicate when the data transfer is completed in the QSPI memory space by setting QSPI_CR.LASTXFR. The end of the instruction frame is indicated when QSPI_SR.INSTRE rises.

The following figure illustrates instruction transmission management.

#### 49.6.5.3 Read Memory Transfer

The user can access the data of the serial memory by sending an instruction with QSPI_IFR.DATAEN = 1 and QSPI_IFR.TFRTYP = 1.

In this mode, the QSPI is able to read data at random address into the serial Flash memory, allowing the CPU to execute code directly from it (XIP execute-in-place).

In order to fetch data, the user must first configure the instruction frame by writing the QSPI_IFR. Then data can be read at any address in the QSPI address space mapping. The address of the system bus read accesses match the address of the data inside the serial Flash memory.

When Fetch mode is enabled, several instruction frames can be sent before writing QSPI_CR.LASTXFR. Each time the system bus read accesses become nonsequential (addresses are not consecutive), a new instruction frame is sent with the corresponding address.

#### 49.6.5.4 Continuous Read Mode

The QSPI is compatible with the Continuous Read mode which is implemented in some serial Flash memories.

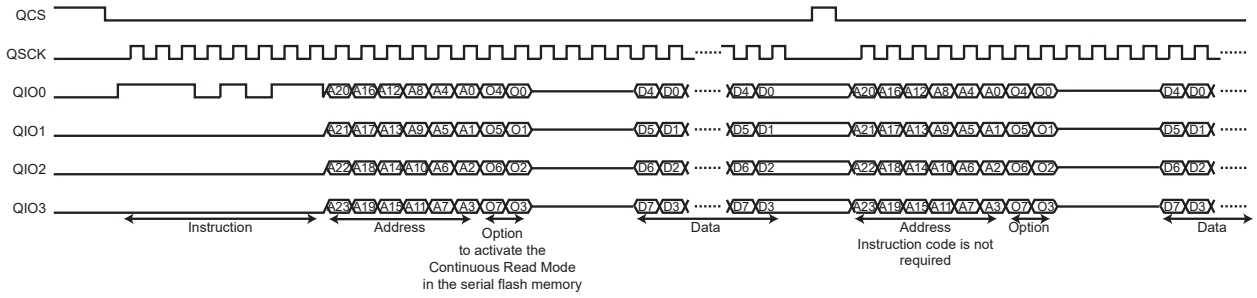
In Continuous Read mode, the instruction overhead is reduced by excluding the instruction code from the instruction frame. When the Continuous Read mode is activated in a serial Flash memory by a specific option code, the instruction code is stored in the memory. For the next instruction frames, the instruction code is not required as the memory uses the stored one.

In the QSPI, Continuous Read mode is used when reading data from the memory (QSPI_IFR.TFRTYP = 1). The addresses of the system bus read accesses are often nonsequential and this leads to many instruction frames that have the same instruction code. By disabling the send of the instruction code, the Continuous Read mode reduces the access time of the data.

To be functional, this mode must be enabled in both the QSPI and the serial Flash memory. The Continuous Read mode is enabled in the QSPI by writing CRM to '1' in the QSPI_IFR (TFRTYP must equal 1). The Continuous Read mode is enabled in the serial Flash memory by sending a specific option code.

**⚠ CAUTION** If the Continuous Read mode is not supported by the serial Flash memory or disabled, CRM bit must not be written to '1', otherwise data read out of the serial Flash memory is unpredictable.

**Figure 49-9. Continuous Read Mode**




#### 49.6.5.5 Instruction Frame Transmission Examples

All waveforms in the following examples describe SPI transfers in SPI Clock mode 0 (QSPI_SCR.CPOL = 0 and QSPI_SCR.CPHA = 0; see section Serial Clock Phase and Polarity).

All system bus accesses described below refer to the system bus address phase. System bus wait cycles and system bus data phases are not shown.

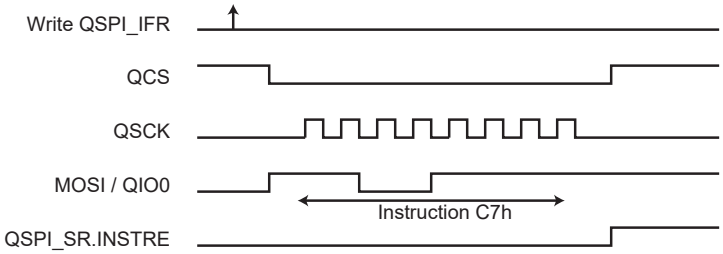
Example 1:

Instruction in Single-bit SPI, without address, without option, without data.

Command: CHIP ERASE (C7h).

- Write 0x0000_00C7 in QSPI_ICR.
- Write 0x0000_0010 in QSPI_IFR.
- Wait for QSPI_SR.INSTRE to rise.

**Figure 49-10. Instruction Transmission Waveform 1**



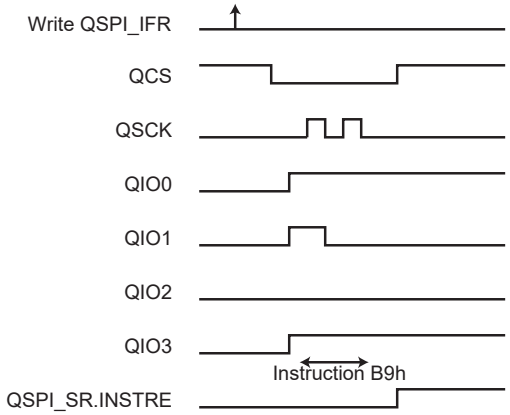

Example 2:

Instruction in Quad SPI, without address, without option, without data.

Command: POWER DOWN (B9h)

- Write 0x0000_00B9 in QSPI_ICR.
- Write 0x0000_0016 in QSPI_IFR.
- Wait for QSPI_SR.INSTRE to rise.

**Figure 49-11. Instruction Transmission Waveform 2**



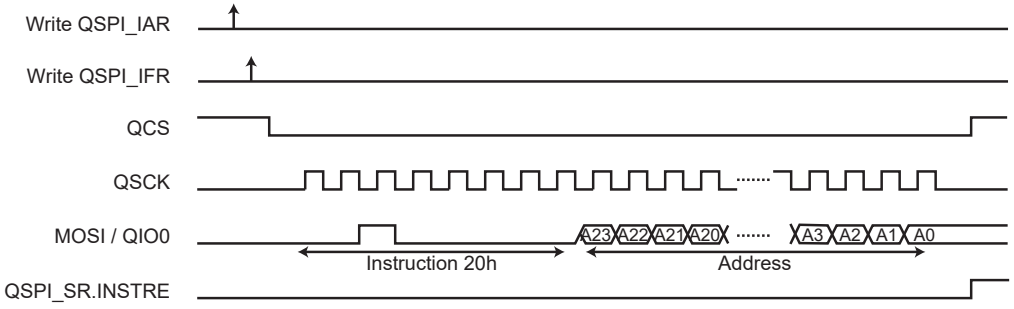

Example 3:

Instruction in Single-bit SPI, with address in Single-bit SPI, without option, without data.

Command: BLOCK ERASE (20h)

- Write the address (of the block to erase) in QSPI_AR.
- Write 0x0000_0020 in QSPI_ICR.
- Write 0x0000_0030 in QSPI_IFR.
- Wait for QSPI_SR.INSTRE to rise.

**Figure 49-12. Instruction Transmission Waveform 3**



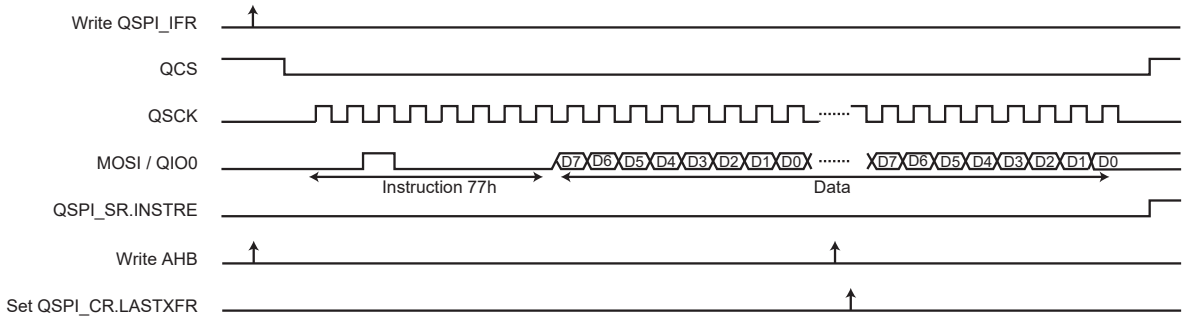
Example 4:

Instruction in Single-bit SPI, without address, without option, with data write in Single-bit SPI.

Command: SET BURST (77h)

- Write 0x0000_0077 in QSPI_ICR.
- Write 0x0000_2090 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Write data in the system bus memory space (0x90000000/0x98000000/0xD0000000/0xD8000000). The address of system bus write accesses is not used.
- Write a '1' to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

**Figure 49-13. Instruction Transmission Waveform 4**



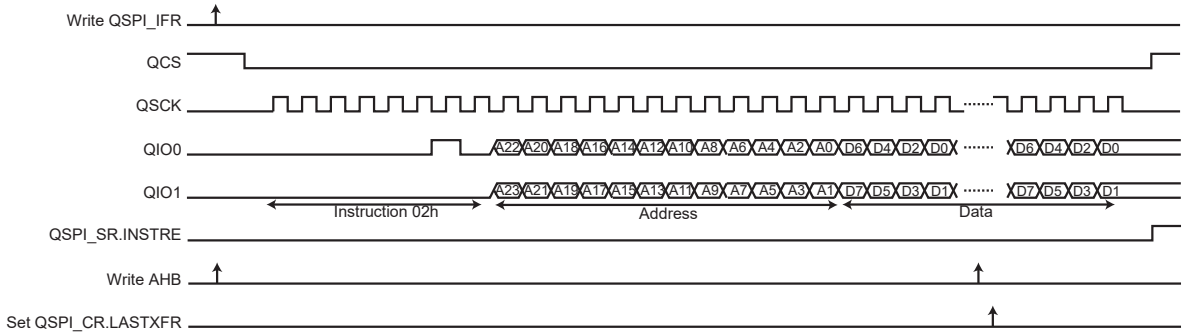

Example 5:

Instruction in Single-bit SPI, with address in Dual SPI, without option, with data write in Dual SPI.

Command: BYTE/PAGE PROGRAM (02h)

- Write 0x0000_0002 in QSPI_ICR.
- Write 0x0000_30B3 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Write data in the QSPI system bus memory space (0x90000000/0x98000000/0xD0000000/0xD8000000).
  The address of the first system bus write access is sent in the instruction frame.
  The address of the next system bus write accesses is not used.
- Write a '1' to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

**Figure 49-14. Instruction Transmission Waveform 5**



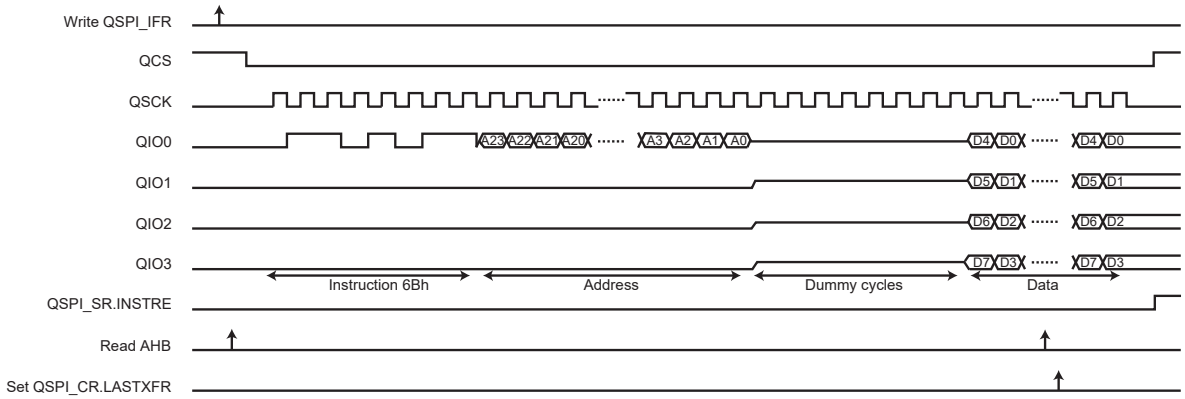
Example 6:

Instruction in Single-bit SPI, with address in Single-bit SPI, without option, with data read in Quad SPI, with eight dummy cycles.

Command: QUAD_OUTPUT READ ARRAY (6Bh)

- Write 0x0000_006B in QSPI_ICR.
- Write 0x0008_10B2 in QSPI_IFR.
- Read QSPI_IR (dummy read) to synchronize system bus accesses.
- Read data in the QSPI system bus memory space (0x90000000/0x98000000/0xD0000000/0xD8000000).
  The address of the first system bus read access is sent in the instruction frame.
  The address of the next system bus read accesses is not used.
- Write a '1' to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

**Figure 49-15. Instruction Transmission Waveform 6**



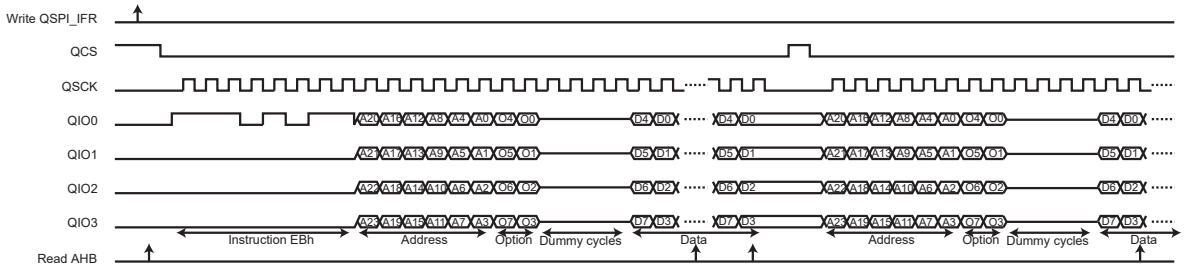

Example 7:

Instruction in Single-bit SPI, with address and option in Quad SPI, with data read in Quad SPI, with four dummy cycles, with fetch and continuous read.

Command: FAST READ QUAD I/O (EBh) - 8-BIT OPTION (0x30h)

- Write 0x0030_00EB in QSPI_ICR.
- Write 0x0004_33F4 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Read data in the QSPI system bus memory space (0x90000000/0x98000000/0xD0000000/0xD8000000).
  Fetch is enabled, the address of the system bus read accesses is always used.
- Write a '1' to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

**Figure 49-16. Instruction Transmission Waveform 7**



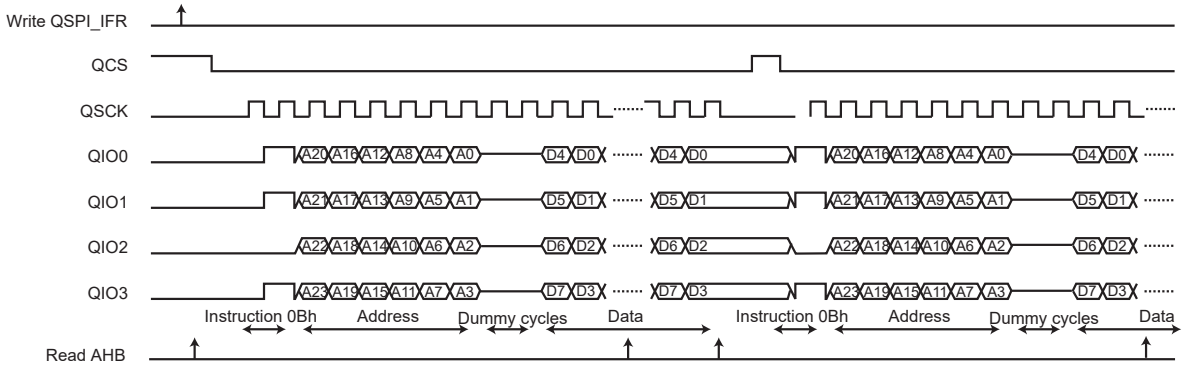

Example 8:

Instruction in Quad SPI, with address in Quad SPI, without option, with data read in Quad SPI, with two dummy cycles, with fetch.

Command: HIGH-SPEED READ (0Bh)

- Write 0x0000_000B in QSPI_ICR.
- Write 0x0002_20B6 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Read data in the QSPI system bus memory space (0x90000000/0x98000000/0xD0000000/0xD8000000).
  Fetch is enabled, the address of the system bus read accesses is always used.
- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

**Figure 49-17. Instruction Transmission Waveform 8**




Example 9:

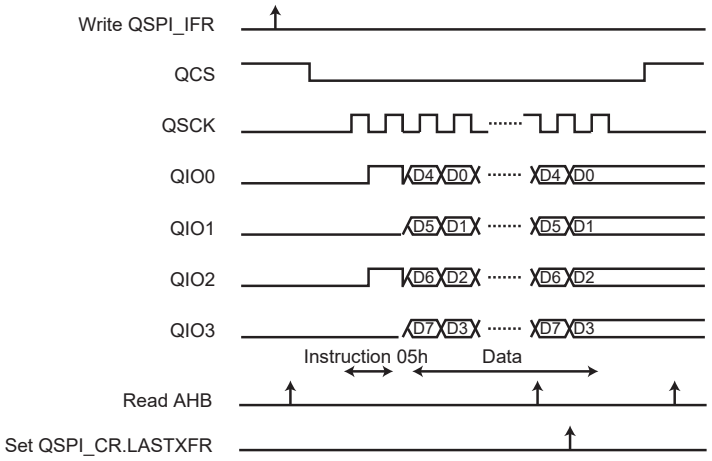
Instruction in Quad SPI, without address, without option, with data read in Quad SPI, without dummy cycles, without fetch.

Command: HIGH-SPEED READ (05h)

- Write 0x0000_0005 in QSPI_ICR.
- Write 0x0000_0096 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Read data in the QSPI system bus memory space (0x90000000/0x98000000/0xD0000000/0xD8000000).
  Fetch is disabled.
- Write a ‘1’ to QSPI_CR.LASTXFR.

- Wait for QSPI_SR.INSTRE to rise.

**Figure 49-18. Instruction Transmission Waveform 9**



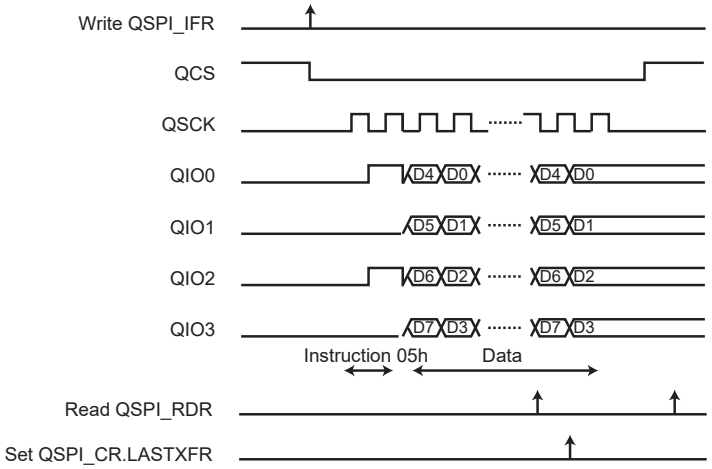

Example 10:

Instruction in Quad SPI, without address, without option, with data read in Quad SPI, without dummy cycles, without fetch, read launched through APB interface.

Command: HIGH-SPEED READ (05h)

- Set SMRM to ‘1’ in QSPI_MR
- Write 0x0000_0005 in QSPI_ICR.
- Write 0x0100_0096 in QSPI_IFR (will start the transfer).
- Wait flag RDRF and Read data in the QSPI_RDR register
  Fetch is disabled.
- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

**Figure 49-19. Instruction Transmission Waveform 10**




### 49.6.6 Scrambling/Unscrambling Function

The scrambling/unscrambling function cannot be performed on devices other than memories. Data is scrambled when written to memory and unscrambled when data is read.

The external data lines can be scrambled in order to prevent intellectual property data located in off-chip memories from being easily recovered by analyzing data at the package pin level of either the microcontroller or the QSPI slave device (e.g., memory).

The scrambling/unscrambling function can be enabled by writing a '1' to the SCREN bit in the QSPI Scrambling Mode Register (QSPI_SMR).

The scrambling and unscrambling are performed on-the-fly without impacting the throughput.

The scrambling method depends on the user-configurable user scrambling key (field USRK) in the QSPI Scrambling Key Register (QSPI_SKR). QSPI_SKR is only accessible in Write mode.

If QSPI_SMR.RVDIS is written to '0', the scrambling/unscrambling algorithm includes the user scrambling key plus a random value depending on device processing characteristics. Data scrambled by a given microcontroller cannot be unscrambled by another.

If QSPI_SMR.RVDIS is written to '1', the scrambling/unscrambling algorithm includes only the user scrambling key. No random value is part of the key.

The user scrambling key or the seed for key generation must be securely stored in a reliable nonvolatile memory in order to recover data from the off-chip memory. Any data scrambled with a given key cannot be recovered if the key is lost.

### 49.6.7 Register Write Protection

To prevent any single software error from corrupting QSPI behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the QSPI Write Protection Mode Register (QSPI_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the QSPI Write Protection Status Register (QSPI_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the QSPI_WPSR.

The following registers can be write-protected when WPEN is set in QSPI_WPMR:

- QSPI Mode Register
- QSPI Serial Clock Register
- QSPI Scrambling Mode Register
- QSPI Scrambling Key Register

## 49.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | QSPI_CR | 7:0 | SWRST | | | | | | QSPIDIS | QSPIEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | LASTXFER |
| 0x04 | QSPI_MR | 7:0 | | | CSMODE[1:0] | | SMRM | WDRBT | LLB | SMM |
| | | 15:8 | | | | | NBBITS[3:0] | | | |
| | | 23:16 | DLYBCT[7:0] | | | | | | | |
| | | 31:24 | DLYCS[7:0] | | | | | | | |
| 0x08 | QSPI_RDR | 7:0 | RD[7:0] | | | | | | | |
| | | 15:8 | RD[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | QSPI_TDR | 7:0 | TD[7:0] | | | | | | | |
| | | 15:8 | TD[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | QSPI_SR | 7:0 | | | | | OVRES | TXEMPTY | TDRE | RDRF |
| | | 15:8 | | | | | | INSTRE | CSS | CSR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | QSPIENS |
| 0x14 | QSPI_IER | 7:0 | | | | | OVRES | TXEMPTY | TDRE | RDRF |
| | | 15:8 | | | | | | INSTRE | CSS | CSR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | QSPI_IDR | 7:0 | | | | | OVRES | TXEMPTY | TDRE | RDRF |
| | | 15:8 | | | | | | INSTRE | CSS | CSR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | QSPI_IMR | 7:0 | | | | | OVRES | TXEMPTY | TDRE | RDRF |
| | | 15:8 | | | | | | INSTRE | CSS | CSR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | QSPI_SCR | 7:0 | | | | | | | CPHA | CPOL |
| | | 15:8 | SCBR[7:0] | | | | | | | |
| | | 23:16 | DLYBS[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | QSPI_IAR | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x34 | QSPI_ICR | 7:0 | INST[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | OPT[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x38 | QSPI_IFR | 7:0 | DATAEN | OPTEN | ADDREN | INSTEN | | WIDTH[2:0] | | |
| | | 15:8 | | CRM | TFRTYP[1:0] | | | ADDRL | OPTL[1:0] | |
| | | 23:16 | | | | NBDUM[4:0] | | | | |
| | | 31:24 | | | | | | | | |
| 0x3C ... 0x3F | Reserved | | | | | | | | | |
| 0x40 | QSPI_SMR | 7:0 | | | | | | | RVDIS | SCREN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x44 | QSPI_SKR | 7:0 | USRK[7:0] | | | | | | | |
| | | 15:8 | USRK[15:8] | | | | | | | |
| | | 23:16 | USRK[23:16] | | | | | | | |
| | | 31:24 | USRK[31:24] | | | | | | | |
| 0x48 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | QSPI_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | QSPI_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 49.7.1 QSPI Control Register

**Name:** QSPI_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | LASTXFER |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWRST | | | | | | QSPIDIS | QSPIEN |
| Access | W | | | | | | W | W |
| Reset | – | | | | | | – | – |

**Bit 24 – LASTXFER** Last Transfer

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The chip select is deasserted after the character written in QSPI_TDR.TD has been transferred. |

**Bit 7 – SWRST** QSPI Software Reset
DMA channels are not affected by software reset.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Reset the QSPI. A software-triggered hardware reset of the QSPI interface is performed. |

**Bit 1 – QSPIDIS** QSPI Disable
As soon as QSPIDIS is set, the QSPI finishes its transfer.

All pins are set in Input mode and no data is received or transmitted.

If a transfer is in progress, the transfer is finished before the QSPI is disabled.

If both QSPIEN and QSPIDIS are equal to one when QSPI_CR is written, the QSPI is disabled.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the QSPI. |

**Bit 0 – QSPIEN** QSPI Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the QSPI to transfer and receive data. |

### 49.7.2 QSPI Mode Register

**Name:** QSPI_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bit WPEN is cleared in the QSPI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DLYCS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DLYBCT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | NBBITS[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | CSMODE[1:0] | | SMRM | WDRBT | LLB | SMM |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – DLYCS[7:0]** Minimum Inactive QCS Delay
This field defines the minimum delay between the deactivation and the activation of QCS. The DLYCS time guarantees the slave minimum deselect time.

If DLYCS written to ‘0’, one peripheral clock period is inserted by default.

Otherwise, the following equation determines the delay:

DLYCS = Minimum inactive × $f_{peripheral\ clock}$

**Bits 23:16 – DLYBCT[7:0]** Delay Between Consecutive Transfers
This field defines the delay between two consecutive transfers with the same peripheral without removing the chip select. The delay is always inserted after each transfer and before removing the chip select if needed.

When DLYBCT is written to ‘0’, no delay between consecutive transfers is inserted and the clock keeps its duty cycle over the character transfers. In Serial Memory mode (SMM = 1), DLYBCT must be written to ‘0’ and no delay is inserted.

Otherwise, the following equation determines the delay:

DLYBCT = (Delay Between Consecutive Transfers × $f_{peripheral\ clock}$) / 32

**Bits 11:8 – NBBITS[3:0]** Number Of Bits Per Transfer

| Value | Name | Description |
|---|---|---|
| 0 | 8_BIT | 8 bits for transfer |
| 8 | 16_BIT | 16 bits for transfer |

**Bits 5:4 – CSMODE[1:0]** Chip Select Mode
The CSMODE field determines how the chip select is deasserted

**Note:** This field is forced to LASTXFER when SMM is written to '1'.

| Value | Name | Description |
|---|---|---|
| 0 | NOT_RELOADED | The chip select is deasserted if QSPI_TDR.TD has not been reloaded before the end of the current transfer. |
| 1 | LASTXFER | The chip select is deasserted when the bit LASTXFER is written to '1' and the character written in QSPI_TDR.TD has been transferred. |
| 2 | SYSTEMATICALLY | The chip select is deasserted systematically after each transfer. |

**Bit 3 – SMRM** Serial Memory Register Mode

| Value | Description |
|---|---|
| 0 | Serial Memory registers are written via AHB access. See section Instruction Frame Transmission for details. |
| 1 | Serial Memory registers are written via APB access. See section Instruction Frame Transmission for details. |

**Bit 2 – WDRBT** Wait Data Read Before Transfer
0 (DISABLED): No effect. In SPI mode, a transfer can be initiated whatever the state of the QSPI_RDR is.

1 (ENABLED): In SPI mode, a transfer can start only if the QSPI_RDR is empty, i.e., does not contain any unread data. This mode prevents overrun error in reception.

**Bit 1 – LLB** Local Loopback Enable
0 (DISABLED): Local loopback path disabled.

1 (ENABLED): Local loopback path enabled.

LLB controls the local loopback on the data serializer for testing in SPI mode only. (MISO is internally connected on MOSI).

**Bit 0 – SMM** Serial Memory Mode
0 (SPI): The QSPI is in SPI mode.

1 (MEMORY): The QSPI is in Serial Memory mode.

### 49.7.3 QSPI Receive Data Register

**Name:** QSPI_RDR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RD[15:0]** Receive Data
Data received by the QSPI is stored in this register right-justified. Unused bits read zero.

### 49.7.4 QSPI Transmit Data Register

**Name:** QSPI_TDR
**Offset:** 0x0C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TD[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TD[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 15:0 – TD[15:0]** Transmit Data
Data to be transmitted by the QSPI is stored in this register. Information to be transmitted must be written to the Transmit Data register in a right-justified format.

### 49.7.5 QSPI Status Register

**Name:** QSPI_SR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | QSPIENS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | INSTRE | CSS | CSR |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | TXEMPTY | TDRE | RDRF |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 24 – QSPIENS** QSPI Enable Status

| Value | Description |
|---|---|
| 0 | QSPI is disabled. |
| 1 | QSPI is enabled. |

**Bit 10 – INSTRE** Instruction End Status (cleared on read)

| Value | Description |
|---|---|
| 0 | No instruction end has been detected since the last read of QSPI_SR. |
| 1 | At least one instruction end has been detected since the last read of QSPI_SR. |

**Bit 9 – CSS** Chip Select Status

| Value | Description |
|---|---|
| 0 | The chip select is asserted. |
| 1 | The chip select is not asserted. |

**Bit 8 – CSR** Chip Select Rise (cleared on read)

| Value | Description |
|---|---|
| 0 | No chip select rise has been detected since the last read of QSPI_SR. |
| 1 | At least one chip select rise has been detected since the last read of QSPI_SR. |

**Bit 3 – OVRES** Overrun Error Status (cleared on read)
An overrun occurs when QSPI_RDR is loaded at least twice from the serializer since the last read of the QSPI_RDR.

| Value | Description |
|---|---|
| 0 | No overrun has been detected since the last read of QSPI_SR. |
| 1 | At least one overrun error has occurred since the last read of QSPI_SR. |

**Bit 2 – TXEMPTY** Transmission Registers Empty (cleared by writing QSPI_TDR)

| Value | Description |
|---|---|
| 0 | As soon as data is written in QSPI_TDR. |
| 1 | QSPI_TDR and the internal shifter are empty. If a transfer delay has been defined, TXEMPTY is set after the completion of such delay. |

**Bit 1 – TDRE** Transmit Data Register Empty (cleared by writing QSPI_TDR)
TDRE equals zero when the QSPI is disabled or at reset. The QSPI enable command sets this bit to one.

| Value | Description |
|---|---|
| 0 | Data has been written to QSPI_TDR and not yet transferred to the serializer. |
| 1 | The last data written in the QSPI_TDR has been transferred to the serializer. |

**Bit 0 – RDRF** Receive Data Register Full (cleared by reading QSPI_RDR)

| Value | Description |
|---|---|
| 0 | No data has been received since the last read of QSPI_RDR. |
| 1 | Data has been received and the received data has been transferred from the serializer to QSPI_RDR since the last read of QSPI_RDR. |

### 49.7.6 QSPI Interrupt Enable Register

**Name:** QSPI_IER
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | INSTRE | CSS | CSR |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | TXEMPTY | TDRE | RDRF |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

**Bit 10 – INSTRE** Instruction End Interrupt Enable

**Bit 9 – CSS** Chip Select Status Interrupt Enable

**Bit 8 – CSR** Chip Select Rise Interrupt Enable

**Bit 3 – OVRES** Overrun Error Interrupt Enable

**Bit 2 – TXEMPTY** Transmission Registers Empty Enable

**Bit 1 – TDRE** Transmit Data Register Empty Interrupt Enable

**Bit 0 – RDRF** Receive Data Register Full Interrupt Enable

### 49.7.7 QSPI Interrupt Disable Register

**Name:** QSPI_IDR
**Offset:** 0x18
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | INSTRE | CSS | CSR |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | TXEMPTY | TDRE | RDRF |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

**Bit 10 – INSTRE** Instruction End Interrupt Disable

**Bit 9 – CSS** Chip Select Status Interrupt Disable

**Bit 8 – CSR** Chip Select Rise Interrupt Disable

**Bit 3 – OVRES** Overrun Error Interrupt Disable

**Bit 2 – TXEMPTY** Transmission Registers Empty Disable

**Bit 1 – TDRE** Transmit Data Register Empty Interrupt Disable

**Bit 0 – RDRF** Receive Data Register Full Interrupt Disable

### 49.7.8 QSPI Interrupt Mask Register

**Name:** QSPI_IMR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | INSTRE | CSS | CSR |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | TXEMPTY | TDRE | RDRF |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 10 – INSTRE** Instruction End Interrupt Mask

**Bit 9 – CSS** Chip Select Status Interrupt Mask

**Bit 8 – CSR** Chip Select Rise Interrupt Mask

**Bit 3 – OVRES** Overrun Error Interrupt Mask

**Bit 2 – TXEMPTY** Transmission Registers Empty Mask

**Bit 1 – TDRE** Transmit Data Register Empty Interrupt Mask

**Bit 0 – RDRF** Receive Data Register Full Interrupt Mask

### 49.7.9 QSPI Serial Clock Register

**Name:** QSPI_SCR
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bit WPEN is cleared in the QSPI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DLYBS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SCBR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CPHA | CPOL |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bits 23:16 – DLYBS[7:0]** Delay Before QSCK
This field defines the delay from QCS valid to the first valid QSCK transition.

When DLYBS equals zero, the QCS valid to QSCK transition is 1/2 the QSCK clock period.

Otherwise, the following equation determines the delay:

DLYBS = Delay Before QSCK × $f_{peripheral\ clock}$

**Bits 15:8 – SCBR[7:0]** Serial Clock Baud Rate
The QSPI uses a modulus counter to derive the QSCK baud rate from the peripheral clock. The baud rate is selected by writing a value from 0 to 255 in the SCBR field. The following equation determines the QSCK baud rate:

SCBR = ($f_{peripheral\ clock}$ / QSCK Baudrate) - 1

**Bit 1 – CPHA** Clock Phase
CPHA determines which edge of QSCK causes data to change and which edge causes data to be captured. CPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.

| Value | Description |
|---|---|
| 0 | Data is captured on the leading edge of QSCK and changed on the following edge of QSCK. |

| Value | Description |
|---|---|
| 1 | Data is changed on the leading edge of QSCK and captured on the following edge of QSCK. |

**Bit 0 – CPOL** Clock Polarity
CPOL is used to determine the inactive state value of the serial clock (QSCK). It is used with CPHA to produce the required clock/data relationship between master and slave devices.

| Value | Description |
|---|---|
| 0 | The inactive state value of QSCK is logic level zero. |
| 1 | The inactive state value of QSCK is logic level one. |

### 49.7.10 QSPI Instruction Address Register

**Name:** QSPI_IAR
**Offset:** 0x30
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** Address
Address to send to the serial Flash memory in the instruction frame.

### 49.7.11 QSPI Instruction Code Register

**Name:** QSPI_ICR
**Offset:** 0x34
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | OPT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | INST[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – OPT[7:0]** Option Code
Option code to send to the serial Flash memory.

**Bits 7:0 – INST[7:0]** Instruction Code
Instruction code to send to the serial Flash memory.

### 49.7.12 QSPI Instruction Frame Register

**Name:** QSPI_IFR
**Offset:** 0x38
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | NBDUM[4:0] | | | | |
| Access | | | | | | | | |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | CRM | TFRTYP[1:0] | | | ADDRL | OPTL[1:0] | |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATAEN | OPTEN | ADDREN | INSTEN | | WIDTH[2:0] | | |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bits 20:16 – NBDUM[4:0]** Number Of Dummy Cycles
The NBDUM field defines the number of dummy cycles required by the serial Flash memory before data transfer.

**Bit 14 – CRM** Continuous Read Mode
0 (DISABLED): Continuous Read mode is disabled.

1 (ENABLED): Continuous Read mode is enabled.

**Bits 13:12 – TFRTYP[1:0]** Data Transfer Type

| Value | Name | Description |
|---|---|---|
| 0 | TRSFR_READ | Read transfer from the serial memory.<br>Scrambling is not performed.<br>Read at random location (fetch) in the serial Flash memory is not possible. |
| 1 | TRSFR_READ_MEMORY | Read data transfer from the serial memory.<br>If enabled, scrambling is performed.<br>Read at random location (fetch) in the serial Flash memory is possible. |

| Value | Name | Description |
|---|---|---|
| 2 | TRSFR_WRITE | Write transfer into the serial memory.<br>Scrambling is not performed. |
| 3 | TRSFR_WRITE_MEMORY | Write data transfer into the serial memory.<br>If enabled, scrambling is performed. |

**Bit 10 – ADDRL** Address Length
The ADDRL bit determines the length of the address.

0 (24_BIT): The address is 24 bits long.

1 (32_BIT): The address is 32 bits long.

**Bits 9:8 – OPTL[1:0]** Option Code Length
The OPTL field determines the length of the option code. The value written in OPTL must be consistent with the value written in the field WIDTH. For example, OPTL = 0 (1-bit option code) is not consistent with WIDTH = 6 (option code sent with QuadSPI protocol, thus the minimum length of the option code is 4 bits).

| Value | Name | Description |
|---|---|---|
| 0 | OPTION_1BIT | The option code is 1 bit long. |
| 1 | OPTION_2BIT | The option code is 2 bits long. |
| 2 | OPTION_4BIT | The option code is 4 bits long. |
| 3 | OPTION_8BIT | The option code is 8 bits long. |

**Bit 7 – DATAEN** Data Enable

| Value | Description |
|---|---|
| 0 | No data is sent/received to/from the serial Flash memory. |
| 1 | Data is sent/received to/from the serial Flash memory. |

**Bit 6 – OPTEN** Option Enable

| Value | Description |
|---|---|
| 0 | The option is not sent to the serial Flash memory. |
| 1 | The option is sent to the serial Flash memory. |

**Bit 5 – ADDREN** Address Enable

| Value | Description |
|---|---|
| 0 | The transfer address is not sent to the serial Flash memory. |
| 1 | The transfer address is sent to the serial Flash memory. |

**Bit 4 – INSTEN** Instruction Enable

| Value | Description |
|---|---|
| 0 | The instruction is not sent to the serial Flash memory. |
| 1 | The instruction is sent to the serial Flash memory. |

**Bits 2:0 – WIDTH[2:0]** Width of Instruction Code, Address, Option Code and Data

| Value | Name | Description |
|---|---|---|
| 0 | SINGLE_BIT_SPI | Instruction: Single-bit SPI / Address-Option: Single-bit SPI / Data: Single-bit SPI |
| 1 | DUAL_OUTPUT | Instruction: Single-bit SPI / Address-Option: Single-bit SPI / Data: Dual SPI |
| 2 | QUAD_OUTPUT | Instruction: Single-bit SPI / Address-Option: Single-bit SPI / Data: Quad SPI |
| 3 | DUAL_IO | Instruction: Single-bit SPI / Address-Option: Dual SPI / Data: Dual SPI |
| 4 | QUAD_IO | Instruction: Single-bit SPI / Address-Option: Quad SPI / Data: Quad SPI |
| 5 | DUAL_CMD | Instruction: Dual SPI / Address-Option: Dual SPI / Data: Dual SPI |
| 6 | QUAD_CMD | Instruction: Quad SPI / Address-Option: Quad SPI / Data: Quad SPI |

### 49.7.13 QSPI Scrambling Mode Register

**Name:** QSPI_SMR
**Offset:** 0x40
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bit WPEN is cleared in the QSPI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RVDIS | SCREN |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – RVDIS** Scrambling/Unscrambling Random Value Disable

| Value | Description |
|---|---|
| 0 | The scrambling/unscrambling algorithm includes the user scrambling key plus a random value that may differ between devices. |
| 1 | The scrambling/unscrambling algorithm includes only the user scrambling key. |

**Bit 0 – SCREN** Scrambling/Unscrambling Enable
0 (DISABLED): The scrambling/unscrambling is disabled.

1 (ENABLED): The scrambling/unscrambling is enabled.

### 49.7.14 QSPI Scrambling Key Register

**Name:** QSPI_SKR
**Offset:** 0x44
**Reset:** –
**Property:** Write-only

This register can only be written if bit WPEN is cleared in the QSPI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | USRK[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | USRK[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | USRK[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | USRK[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – USRK[31:0]** User Scrambling Key

### 49.7.15 QSPI Write Protection Mode Register

**Name:** QSPI_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | WPEN |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x515350 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See section Register Write Protection for the list of registers that can be protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x515350 (QSP in ASCII) |
| 1 | Enables the write protection if WPKEY corresponds to 0x515350 (QSP in ASCII) |

### 49.7.16 QSPI Write Protection Status Register

**Name:** QSPI_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 15:8 – WPVSRC[7:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of the QSPI_WPSR. |
| 1 | A write protection violation has occurred since the last read of the QSPI_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 50. Secure Digital MultiMedia Card Controller (SDMMC)

## 50.1 Description

The Secure Digital MultiMedia Card Controller (SDMMC) supports the embedded MultiMedia Card (e.MMC) Specification V4.51, the SD Memory Card Specification V3.0, and the SDIO V3.0 specification. It is compliant with the SD Host Controller Standard V3.0 specification.

The SDMMC includes the register set defined in the “SD Host Controller Simplified Specification V3.00” and additional registers to manage e.MMC devices, sampling tuning procedure, PAD calibrationand enhanced features.

The SDMMC is clocked by three asynchronous clocks and requires the PMC to be configured first.

## 50.2 Embedded Characteristics

- Compatible with SD Host Controller Standard Specification Version 3.00
- Compatible with MultiMedia Card Specification Version V4.51
- Compatible with SD Memory Card Specification Version 3.00
- Compatible with SDIO Specification Version 3.00
- Support for 1-bit/ 4-bit SD/SDIO Devices
- Support for 1-bit/4-bit/8-bit e.MMC Devices
- Support for SD/SDIO Default Speed (Maximum SDCLK Frequency = 25 MHz)
- Support for SD/SDIO High Speed (Maximum SDCLK Frequency = 50 MHz)
- Support for SD/SDIO UHS-I SDR12 (Maximum SDCLK Frequency = 25 MHz)
- Support for SD/SDIO UHS-I SDR25 (Maximum SDCLK Frequency = 50 MHz)
- Support for SD/SDIO UHS-I SDR50 (Maximum SDCLK Frequency = 100 MHz)
- Support for SD/SDIO UHS-I SDR104 (Maximum SDCLK Frequency = 120 MHz)
- Support for SD/SDIO UHS-I DDR50 (Maximum SDCLK Frequency = 50 MHz)
- Support for SDSC, SDHC and SDXC
- Support for MMC/e.MMC Default Speed (Maximum SDCLK Frequency = 26 MHz)
- Support for MMC/e.MMC High Speed (Maximum SDCLK Frequency = 52 MHz)
- Support for e.MMC High Speed DDR (Maximum SDCLK Frequency = 52 MHz)
- Support for e.MMC HS200 (Maximum SDCLK Frequency = 120 MHz)
- e.MMC Boot Operation Mode Support
- Support for Block Size from 1 to 512 Bytes
- Support for Stream, Block and Multiblock Data Read and Write
  - Advanced DMA and SDMA Capability
- Internal 1024-byte Dual Port RAM
- Support for both Synchronous and Asynchronous Abort
- Supports for SDIO Card Interrupt

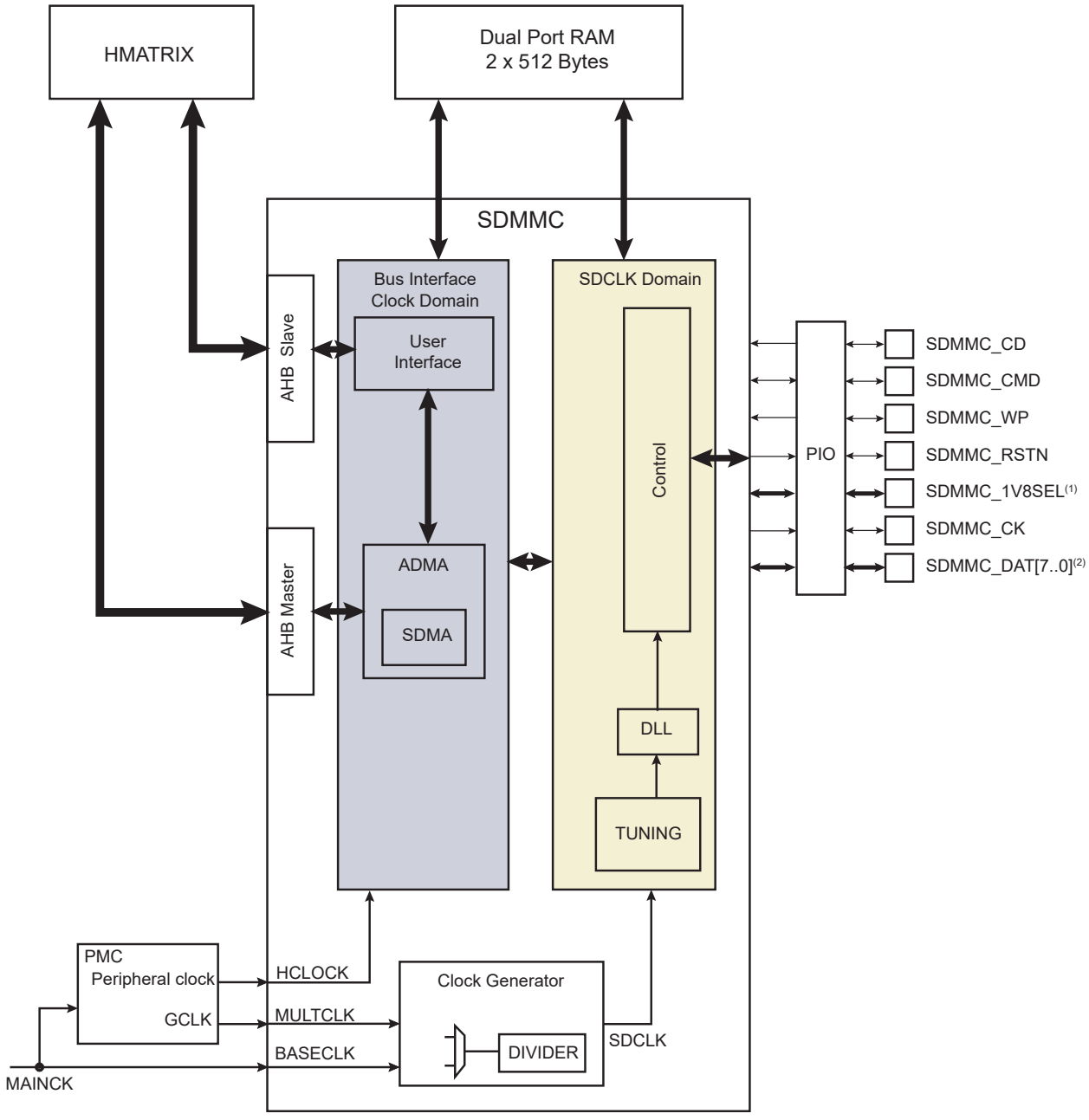
## 50.3 Reference Documents

**Table 50-1. Reference Documents**

| Name | Link |
|---|---|
| SD Host Controller Simplified Specification V3.00 | https://www.sdcard.org |
| SDIO Simplified Specification V3.00 | |
| Physical Layer Simplified Specification V3.01 | |
| Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51 | http://www.jedec.org |

## 50.4 Block Diagram

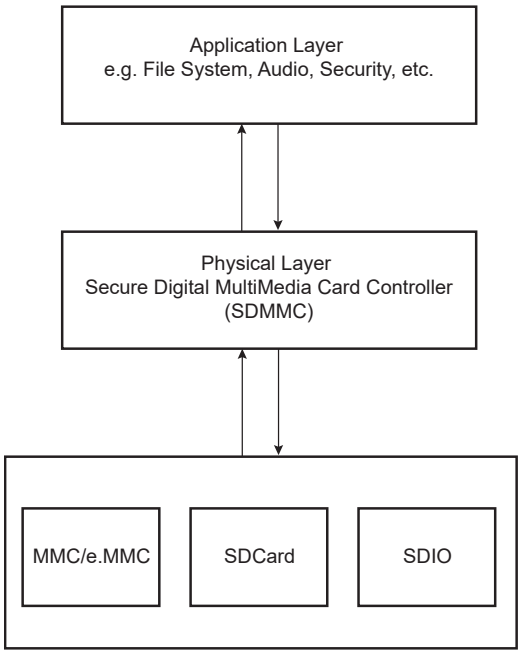
**Figure 50-1. Block Diagram**




Notes: 1. SDMMC_1V8SEL not available on SDMMC1.
2. Limited to SDMMC_DAT[3..0] on SDMMC1.

## 50.5 Application Block Diagram

**Figure 50-2. Application Block Diagram**




## 50.6 Pin Name List

**Table 50-2. I/O Lines Description for 8-bit Configuration**

| Pin Name(1) | Pin Description | Type |
|---|---|---|
| SDMMC_CD | SDCard / SDIO / e.MMC Card Detect | Input |
| SDMMC_CMD | SDCard / SDIO / e.MMC Command/Response Line | I/O |
| SDMMC_WP | SDCard Connector Write Protect Signal | Input |
| SDMMC_RSTN | e.MMC Reset Signal | Output |
| SDMMC_1V8SEL | SDCard Signal Voltage Selection | Output |
| SDMMC_CK | SDCard / SDIO / e.MMC Clock Signal | Output |
| SDMMC_DAT[7..0] | SDCard / SDIO / e.MMC Data Lines | I/O |

Notes: 1. When several SDMMCs are embedded in a product, SDMMC_CK refers to SDMMCx_CK, SDMMC_CMD to SDMMCx_CMD, SDMMC_DATy to SDMMCx_DATy, SDMMC_WP to SDMMCx_WP, SDMMC_1V8SEL to SDMMCx_1V8SEL, SDMMC_CD to SDMMCx_CD and SDMMC_RSTN to SDMMCx_RSTN.

## 50.7 Product Dependencies

### 50.7.1 I/O Lines

The pins used for interfacing the Secure Digital MultiMedia Card (SDMMC) Controller are multiplexed with PIO lines. The programmer must first program the PIO controller to assign the peripheral functions to SDMMC pins.

### 50.7.2 Power Management

The SDMMC is clocked through the Power Management Controller (PMC), so the programmer must first configure the PMC to enable the SDMMC clocks.

### 50.7.3 Interrupt Sources

The SDMMC has an interrupt line connected to the interrupt controller.

Handling the SDMMC interrupt requires programming the interrupt controller before configuring the SDMMC.

## 50.8 SD/SDIO Operating Mode

The SDMMC is fully compliant with the “SD Host Controller Simplified Specification V3.00” for SD/SDIO devices. See this specification for the SDMMC configuration.

See “Physical Layer Simplified Specification V3.01” and “SDIO Simplified Specification V3.00” for SD/SDIO management.

## 50.9 e.MMC Operating Mode

The SDMMC supports management of e.MMC devices. As the “SD Host Controller Simplified Specification V3.00” does not apply to e.MMC devices, some registers have been added to those described in this specification in order to manage e.MMC devices. Most of the registers described in the “SD Host Controller Simplified Specification V3.00” must be used for e.MMC management, but e.MMC-specific features are managed using SDMMC_MC1R and SDMMC_MC2R.

### 50.9.1 Boot Operation Mode

In Boot Operation mode, the processor can read boot data from the e.MMC device by keeping the CMD line low after poweron before issuing the CMD1. The data can be read from either one of the boot partitions or the user area according to BOOT_PARTITION_ENABLE in the Extended CSD register (see “Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51” ).

#### 50.9.1.1 Boot Procedure, Processor Mode

1. Configure the SDMMC:
   a. Set the data bus width using SDMMC_HC1R.DW and SDMMC_HC1R.EXTDW according to the BOOT_BUS_WIDTH in the Extended CSD Register (see “Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51” ).

   b. Select the speed mode (using SDMMC_HC1R.HSEN or SDMMC_MC1R.DDR) according to BOOT_MODE in the Extended CSD Register.

   c. Set the SDCLK frequency according to the selected speed mode.

   d. If the Boot Acknowledge is sent by the e.MMC device (BOOT_ACK = 1 in the Extended CSD Register), set the Boot Acknowledge Enable to ‘1’ (SDMMC_MC1R.BOOTA = 1).

e. Enable the interrupt on Boot Acknowledge Received (SDMMC_NISTER.BOOTAR = 1 and SDMMC_NISIER.BOOTAR = 1).

f. Set the e.MMC Command Type to BOOT (SDMMC_MC1R.CMDTYP = 3)

g. Set SDMMC_TMR to read multiple blocks for the e.MMC device (SDMMC_TMR.MSBSEL = 1 and SDMMC_TMR.DTDSEL = 1).

h. Select the NonDMA transfer (SDMMC_TMR.DMAEN = 0).

i. Optional: select the Auto CMD method (using SDMMC_TMR.ACMDEN).

j. Set the block size to 512 bytes (SDMMC_BSR.BLKSIZE = 512).

k. Set the required number of read blocks (using SDMMC_BCR.BLKCNT). SDMMC_TMR.BCEN must be set to '1'.

2. Write SDMMC_CR = 20(hexa) to set the e.MMC in Boot Operation mode.
3. Wait for interrupt on Boot Acknowledge Received (BOOTAR).
4. The user can copy the boot data sequentially as soon as the BRDRDY flag is asserted.
5. When the data transfer is completed, the boot operation must be terminated by setting SDMMC_MC2R.ABOOT to '1'.

#### 50.9.1.2 Boot Procedure, SDMA Mode

1. Configure SDMMC:
   a. Set the data bus width using SDMMC_HC1R.DW and SDMMC_HC1R.EXTDW according to BOOT_BUS_WIDTH in the Extended CSD Register (see "Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51" ).

   b. Select the speed mode (SDMMC_HC1R.HSEN or SDMMC_MC1R.DDR) according to BOOT_MODE in the Extended CSD Register.

   c. Set the SDCLK frequency according to the selected speed mode.

   d. If the Boot Acknowledge is sent by the e.MMC device (BOOT_ACK = 1 in the Extended CSD Register), set the Boot Acknowledge Enable to 1 (SDMMC_MC1R.BOOTA = 1).

   e. Enable interrupt on Boot Acknowledge Received (SDMMC_NISTER.BOOTAR = 1 and SDMMC_NISIER.BOOTAR = 1).

   f. Set the e.MMC Command Type to BOOT (SDMMC_MC1R.CMDTYP = 3).

   g. Set SDMMC_TMR to read multiple blocks for the e.MMC device (SDMMC_TMR.MSBSEL = 1 and SDMMC_TMR.TDSEL = 1).

   h. Select the SDMA transfer (SDMMC_TMR.DMAEN = 1 and SDMMC_HC1R.DMASEL = 0).

   i. Write the SDMA system address where the boot data will be copied (SDMMC_SSAR.ADDR).

   j. Optional: select the Auto CMD method (SDMMC_TMR.ACMDEN).
   Note: Auto CMD23 cannot be used with SDMA.

   k. Set the block size to 512 bytes (SDMMC_BSR.BLKSIZE = 512).

   l. Set the required number of read blocks (SDMMC_BCR.BLKCNT). SDMMC_TMR.BCEN must be set to 1.
2. Write SDMMC_CR = 20(hexa) to set the e.MMC in Boot Operation mode.
3. Wait for interrupt on Boot Acknowledge Received (BOOTAR).
4. The user can copy the boot data sequentially as soon as the BRDRDY flag is asserted.

5. When the data transfer is completed, the boot operation must be terminated by setting SDMMC_MC2R.ABOOT to ‘1’.

#### 50.9.1.3 Boot Procedure, ADMA Mode

1. Configure the SDMMC:
   a. Set the data bus width using SDMMC_HC1R.DW and SDMMC_HC1R.EXTDW according to BOOT_BUS_WIDTH in the Extended CSD Register (see “Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51” ).

   b. Select the speed mode (SDMMC_HC1R.HSEN or SDMMC_MC1R.DDR) according to BOOT_MODE in the Extended CSD register.

   c. Set the SDCLK frequency according to the selected speed mode.

   d. If the Boot Acknowledge is sent by the e.MMC device (BOOT_ACK = 1 in the Extended CSD Register), set the Boot Acknowledge Enable to ‘1’ (SDMMC_MC1R.BOOTA = 1).

   e. Enable interrupt on Boot Acknowledge Received (SDMMC_NISTER.BOOTAR = 1 and SDMMC_NISIER.BOOTAR = 1).

   f. Set the e.MMC Command Type to BOOT (SDMMC_MC1R.CMDTYP = 3).

   g. Set SDMMC_TMR to read multiple blocks for the e.MMC device (SDMMC_TMR.MSBSEL = 1 and SDMMC_TMR.DTDSEL = 1).

   h. Select the ADMA transfer (SDMMC_TMR.DMAEN = 1 and SDMMC_HC1R.DMASEL = 2 or 3).

   i. Write the address of the descriptor table in the ADMA system address (SDMMC_ASARx [0..1].ADMASA).

   j. Optional: select the Auto CMD method (SDMMC_TMR.ACMDEN).

   k. Set the block size to 512 bytes (SDMMC_BSR.BLKSIZE = 512).

   l. Set the required number of read blocks (SDMMC_BCR.BLKCNT). SDMMC_TMR.BCEN must be set to ‘1’.
2. Write SDMMC_CR = 20(hexa) to set the e.MMC in Boot Operation Mode.
3. Wait for interrupt on Boot Acknowledge Received (BOOTAR).
4. The user can copy the boot data sequentially as soon as the BRDRDY flag is asserted.
5. When the data transfer is completed, the boot operation must be terminated by setting SDMMC_MC2R.ABOOT to ‘1’.

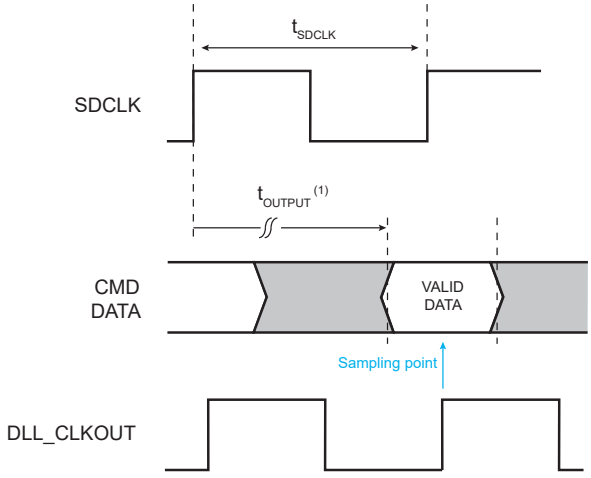
## 50.10 SDR104 / HS200 Tuning

### 50.10.1 DLL and Sampling Point

In SD/SDIO SDR104 mode (SDMMC_HC2R.VS18EN = 1 and SDMMC_HC2R.UHSMS = 3) or e.MMC HS200 mode (HS200EN = $B_{(hexa)}$), a tuning procedure must be performed first in order to adjust the sampling point for read transactions. For more details regarding the basic tuning procedure, see section “Sampling Clock Tuning Procedure” in the “SD Host Controller Simplified Specification V3.00” .

As the position of data and command coming from the device varies, a DLL is used to generate an accurate sampling point (DLL_CLKOUT) (see the figure below).

**Figure 50-3. DLL Sampling Point**

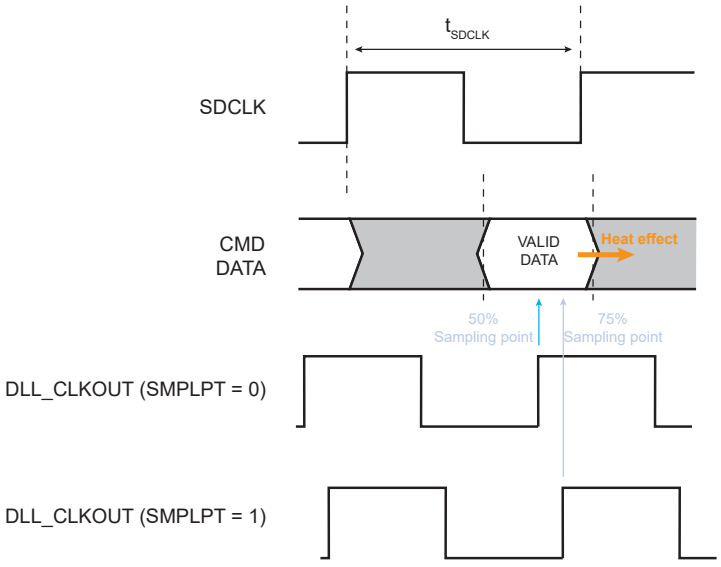



Note: 1. $t_{OUTPUT}$ varies from 0 to 2 x $t_{SDCLK}$

The minimum SDLCK frequency is 100 MHz when SD/SDIO SDR104 or e.MMC HS200 is selected.

The sampling point can be selected to be located at 50% or 75% of the data window to anticipate the effect of the temperature rise. If SDMMC_TUNCR.SMPLPT is cleared, the sampling point is centered (50% of the data window). If SDMMC_TUNCR.SMPLPT is set to ‘1’, the sampling point is set at 75% of the data window (see the figure below).

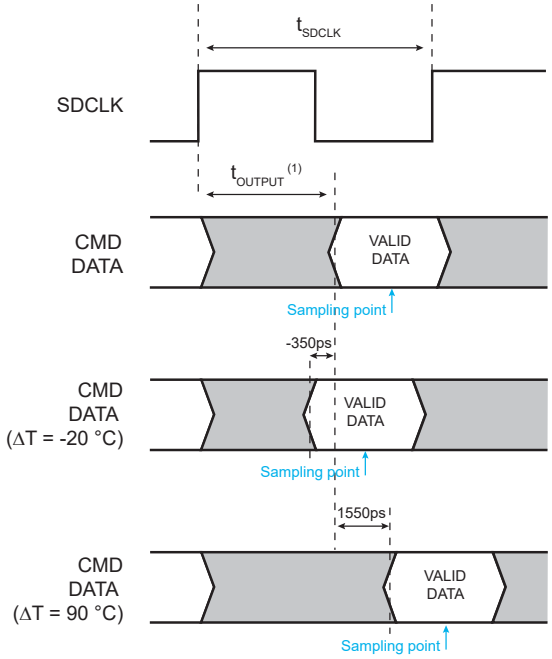
**Figure 50-4. SDR104/HS200 Sampling Point Selection**




### 50.10.2 Retuning Method

Once the data window sampling point has been tuned following the tuning procedure, the data window can be shifted by temperature drift. Thus, the tuning procedure must be applied periodically to adjust the sampling point position. The SDMMC implements a retuning timer which periodically instructs the software to restart the tuning procedure.

**Figure 50-5. Temperature Effect on Data Window**




Note: 1. $t_{OUTPUT}$ varies from 0 to 2 x $t_{SDCLK}$

#### 50.10.2.1 SDMMC Tuning Sequence

The SDMMC tuning sequence must only be done when SD/SDIO SDR104 or e.MMC HS200 is selected and for a 100-MHz SDCLK frequency or higher.

1. Enable the retuning timer (SDMMC_RTC1R.TMREN = 1).
2. Configure the retuning period by setting SDMMC_RTCVR.TCVAL.
3. Set SDMMC_RTISTER.TEVT to ‘1’ so that the TEVT status flag in SDMMC_RTISTR rises each time the retuning timer counter period elapses.
4. Set SDMMC_RTISIER.TEVT to ‘1’ to generate an interrupt on the TEVT status flag assertion (optional).
5. Execute the tuning procedure as defined in “Sampling Clock Tuning Procedure” in the “SD Host Controller Simplified Specification V3.00” .
6. Start the retuning timer count (write SDMMC_RTC2R.RLD to 1). At this step, data can be read by the SDMMC.
7. Each time SDMMC_RTISTR.TEVT is set to ‘1’:
   a. Execute the tuning procedure as defined in “Sampling Clock Tuning Procedure” in the “SD Host Controller Simplified Specification V3.00” before issuing the next command.
   b. Restart the retuning timer count (write SDMMC_RTC2R.RLD to ‘1’).
   c. Resume data reading from the device.

When several instances of SDMMC are implemented in a product, the TEVT status flag of each SDMMC instance can be checked by reading SDMMC_RTSSR.

## 50.11 I/O Calibration

The need for output impedance calibration arises with higher data rates. As the data rate increases, some transmission line effects can occur and lead to the generation of undershoots and overshoots, hence degrading the signal quality.

To avoid these transmission problems, an I/O calibration cell is used to adjust the output impedance to the driven I/Os.

The I/O calibration sequence is mandatory when one of the SD/SDIO UHS-I modes (SDMMC_HC2R.VS18EN = 1) or e.MMC HS200 (HS200EN = $B_{(hexa)}$) is selected. It must be performed periodically to prevent the output impedance drift. Once the calibration is finished, the I/O calibration cell provides two four-bit control words (CALP[3:0] and CALN[3:0] in the Calibration Control register (SDMMC_CALCR)) to tune the output impedance, and thus reach the best transmission performances.

The I/O calibration sequence can be started manually by writing a ‘1’ to SDMMC_CALCR.EN. This bit is cleared automatically at the end of the calibration.

The I/O calibration sequence can also be performed automatically if SDMMC_CALCR.TUNDIS is cleared. In this case, the calibration starts automatically at the beginning of the tuning procedure when writing a ‘1’ to SDMMC_HC2R.EXTUN.

The I/O calibration cell requires a startup time defined by SDMMC_CALCR.CNTVAL. Thus, CNTVAL must be configured prior to start the calibration sequence. If SDMMC_CALCR.ALWYSON is set to ‘1’, the startup time is only required for the first calibration sequence as the analog circuitry is not shut down at the end of the calibration. In order to reduce the power consumption, the analog circuitry can be shut down at the end of the calibration sequence by clearing ALWYSON. In this case, the startup time is performed each time a calibration sequence is started.

## 50.12 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | SDMMC_SSAR | 7:0 | ADDR/ARG2[7:0] | | | | | | | |
| | | 15:8 | ADDR/ARG2[15:8] | | | | | | | |
| | | 23:16 | ADDR/ARG2[23:16] | | | | | | | |
| | | 31:24 | ADDR/ARG2[31:24] | | | | | | | |
| 0x04 | SDMMC_BSR | 7:0 | BLKSIZE[7:0] | | | | | | | |
| | | 15:8 | | BOUNDARY[2:0] | | | | | BLKSIZE[9:8] | |
| 0x06 | SDMMC_BCR | 7:0 | BLKCNT[7:0] | | | | | | | |
| | | 15:8 | BLKCNT[15:8] | | | | | | | |
| 0x08 | SDMMC_ARG1R | 7:0 | ARG1[7:0] | | | | | | | |
| | | 15:8 | ARG1[15:8] | | | | | | | |
| | | 23:16 | ARG1[23:16] | | | | | | | |
| | | 31:24 | ARG1[31:24] | | | | | | | |
| 0x0C | SDMMC_TMR | 7:0 | | | MSBSEL | DTDSEL | ACMDEN[1:0] | | BCEN | DMAEN |
| | | 15:8 | | | | | | | | |
| 0x0E | SDMMC_CR | 7:0 | CMDTYP[1:0] | | DPSEL | CMDICEN | CMDCCEN | | RESPTYP[1:0] | |
| | | 15:8 | | | CMDIDX[5:0] | | | | | |
| 0x10 | SDMMC_RR0 | 7:0 | CMDRESP[7:0] | | | | | | | |
| | | 15:8 | CMDRESP[15:8] | | | | | | | |
| | | 23:16 | CMDRESP[23:16] | | | | | | | |
| | | 31:24 | CMDRESP[31:24] | | | | | | | |
| 0x14 | SDMMC_RR1 | 7:0 | CMDRESP[7:0] | | | | | | | |
| | | 15:8 | CMDRESP[15:8] | | | | | | | |
| | | 23:16 | CMDRESP[23:16] | | | | | | | |
| | | 31:24 | CMDRESP[31:24] | | | | | | | |
| 0x18 | SDMMC_RR2 | 7:0 | CMDRESP[7:0] | | | | | | | |
| | | 15:8 | CMDRESP[15:8] | | | | | | | |
| | | 23:16 | CMDRESP[23:16] | | | | | | | |
| | | 31:24 | CMDRESP[31:24] | | | | | | | |
| 0x1C | SDMMC_RR3 | 7:0 | CMDRESP[7:0] | | | | | | | |
| | | 15:8 | CMDRESP[15:8] | | | | | | | |
| | | 23:16 | CMDRESP[23:16] | | | | | | | |
| | | 31:24 | CMDRESP[31:24] | | | | | | | |
| 0x20 | SDMMC_BDPR | 7:0 | BUFDATA[7:0] | | | | | | | |
| | | 15:8 | BUFDATA[15:8] | | | | | | | |
| | | 23:16 | BUFDATA[23:16] | | | | | | | |
| | | 31:24 | BUFDATA[31:24] | | | | | | | |
| 0x24 | SDMMC_PSR | 7:0 | | | | | | DLACT | CMDINHD | CMDINHC |
| | | 15:8 | | | | | BUFRDEN | BUFWREN | RTACT | WTACT |
| | | 23:16 | DATLL[3:0] | | | | WRPPL | CARDDPL | CARDSS | CARDINS |
| | | 31:24 | | | | | | | | CMDLL |
| 0x28 | SDMMC_HC1R (SD_SDIO) | 7:0 | CARDDSEL | CARDDTL | | DMASEL[1:0] | | HSEN | DW | LEDCTRL |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x28 | SDMMC_HC1R (e.MMC) | 7:0 | | | EXTDW | DMASEL[1:0] | | HSEN | DW | |
| 0x29 | SDMMC_PCR | 7:0 | | | | | | | | SDBPWR |
| 0x2A | SDMMC_BGCR (SD_SDIO) | 7:0 | | | | | INTBG | RWCTRL | CONTR | STPBGR |
| 0x2A | SDMMC_BGCR (e.MMC) | 7:0 | | | | | | | CONTR | STPBGR |
| 0x2B | SDMMC_WCR (SD_SDIO) | 7:0 | | | | | | WKENCREM | WKENCINS | WKENCINT |
| 0x2C | SDMMC_CCR | 7:0 | USDCLKFSEL[1:0] | | CLKGSEL | | | SDCLKEN | INTCLKS | INTCLKEN |
| | | 15:8 | SDCLKFSEL[7:0] | | | | | | | |
| 0x2E | SDMMC_TCR | 7:0 | | | | | DTCVAL[3:0] | | | |
| 0x2F | SDMMC_SRR | 7:0 | | | | | | SWRSTDAT | SWRSTCMD | SWRSTALL |
| 0x30 | SDMMC_NISTR (SD_SDIO) | 7:0 | CREM | CINS | BRDRDY | BWRRDY | DMAINT | BLKGE | TRFC | CMDC |
| | | 15:8 | ERRINT | | | | | | | CINT |
| 0x30 | SDMMC_NISTR (e.MMC) | 7:0 | | | BRDRDY | BWRRDY | DMAINT | BLKGE | TRFC | CMDC |
| | | 15:8 | ERRINT | BOOTAR | | | | | | |
| 0x32 | SDMMC_EISTR (SD_SDIO) | 7:0 | CURLIM | DATEND | DATCRC | DATTEO | CMDIDX | CMDEND | CMDCRC | CMDTEO |
| | | 15:8 | | | | | | | ADMA | ACMD |
| 0x32 | SDMMC_EISTR (e.MMC) | 7:0 | CURLIM | DATEND | DATCRC | DATTEO | CMDIDX | CMDEND | CMDCRC | CMDTEO |
| | | 15:8 | | | | BOOTAE | | | ADMA | ACMD |
| 0x34 | SDMMC_NISTER (SD_SDIO) | 7:0 | CREM | CINS | BRDRDY | BWRRDY | DMAINT | BLKGE | TRFC | CMDC |
| | | 15:8 | | | | | | | | CINT |
| 0x34 | SDMMC_NISTER (e.MMC) | 7:0 | | | BRDRDY | BWRRDY | DMAINT | BLKGE | TRFC | CMDC |
| | | 15:8 | | BOOTAR | | | | | | |
| 0x36 | SDMMC_EISTER (SD_SDIO) | 7:0 | CURLIM | DATEND | DATCRC | DATTEO | CMDIDX | CMDEND | CMDCRC | CMDTEO |
| | | 15:8 | | | | | | | ADMA | ACMD |
| 0x36 | SDMMC_EISTER (e.MMC) | 7:0 | CURLIM | DATEND | DATCRC | DATTEO | CMDIDX | CMDEND | CMDCRC | CMDTEO |
| | | 15:8 | | | | BOOTAE | | | ADMA | ACMD |
| 0x38 | SDMMC_NISIER (SD_SDIO) | 7:0 | CREM | CINS | BRDRDY | BWRRDY | DMAINT | BLKGE | TRFC | CMDC |
| | | 15:8 | | | | | | | | CINT |
| 0x38 | SDMMC_NISIER (e.MMC) | 7:0 | | | BRDRDY | BWRRDY | DMAINT | BLKGE | TRFC | CMDC |
| | | 15:8 | | BOOTAR | | | | | | |
| 0x3A | SDMMC_EISIER (SD_SDIO) | 7:0 | CURLIM | DATEND | DATCRC | DATTEO | CMDIDX | CMDEND | CMDCRC | CMDTEO |
| | | 15:8 | | | | | | | ADMA | ACMD |
| 0x3A | SDMMC_EISIER (e.MMC) | 7:0 | CURLIM | DATEND | DATCRC | DATTEO | CMDIDX | CMDEND | CMDCRC | CMDTEO |
| | | 15:8 | | | | BOOTAE | | | ADMA | ACMD |
| 0x3C | SDMMC_ACESR | 7:0 | CMDNI | | | ACMDIDX | ACMDEND | ACMDCRC | ACMDTEO | ACMD12NE |
| | | 15:8 | | | | | | | | |
| 0x3E | SDMMC_HC2R (SD_SDIO) | 7:0 | SCLKSEL | EXTUN | DRVSEL[1:0] | | VS18EN | UHSMS[2:0] | | |
| | | 15:8 | PVALEN | ASINTEN | | | | | | |
| 0x3E | SDMMC_HC2R (e.MMC) | 7:0 | SCLKSEL | EXTUN | DRVSEL[1:0] | | HS200EN[3:0] | | | |
| | | 15:8 | PVALEN | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x40 | SDMMC_CA0R | 7:0 | TEOCLKU | | TEOCLKF[5:0] | | | | | |
| | | 15:8 | BASECLKF[7:0] | | | | | | | |
| | | 23:16 | SRSUP | SDMASUP | HSSUP | | ADMA2SUP | ED8SUP | MAXBLKL[1:0] | |
| | | 31:24 | SLTYPE[1:0] | | ASINTSUP | SB64SUP | | V18VSUP | V30VSUP | V33VSUP |
| 0x44 | SDMMC_CA1R | 7:0 | | DRVDSUP | DRVCSUP | DRVASUP | | DDR50SUP | SDR104SUP | SDR50SUP |
| | | 15:8 | RTMOD[1:0] | | TSDR50 | | TCNTRT[3:0] | | | |
| | | 23:16 | CLKMULT[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x48 | SDMMC_MCCAR | 7:0 | MAXCUR33V[7:0] | | | | | | | |
| | | 15:8 | MAXCUR30V[7:0] | | | | | | | |
| | | 23:16 | MAXCUR18V[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x4C ... 0x4F | Reserved | | | | | | | | | |
| 0x50 | SDMMC_FERACES | 7:0 | CMDNI | | | ACMDIDX | ACMDEND | ACMDCRC | ACMDTEO | ACMD12NE |
| | | 15:8 | | | | | | | | |
| 0x52 | SDMMC_FEREIS | 7:0 | CURLIM | DATEND | DATCRC | DATTEO | CMDIDX | CMDEND | CMDCRC | CMDTEO |
| | | 15:8 | | | | BOOTAE | | | ADMA | ACMD |
| 0x54 | SDMMC_AESR | 7:0 | | | | | | LMIS | ERRST[1:0] | |
| 0x55 ... 0x57 | Reserved | | | | | | | | | |
| 0x58 | SDMMC_ASAR0 | 7:0 | ADMASA[7:0] | | | | | | | |
| | | 15:8 | ADMASA[15:8] | | | | | | | |
| | | 23:16 | ADMASA[23:16] | | | | | | | |
| | | 31:24 | ADMASA[31:24] | | | | | | | |
| 0x5C ... 0x5F | Reserved | | | | | | | | | |
| 0x60 | SDMMC_PVR0 | 7:0 | SDCLKFSEL[7:0] | | | | | | | |
| | | 15:8 | DRVSEL[1:0] | | | | | CLKGSEL | SDCLKFSEL[9:8] | |
| 0x62 | SDMMC_PVR1 | 7:0 | SDCLKFSEL[7:0] | | | | | | | |
| | | 15:8 | DRVSEL[1:0] | | | | | CLKGSEL | SDCLKFSEL[9:8] | |
| 0x64 | SDMMC_PVR2 | 7:0 | SDCLKFSEL[7:0] | | | | | | | |
| | | 15:8 | DRVSEL[1:0] | | | | | CLKGSEL | SDCLKFSEL[9:8] | |
| 0x66 | SDMMC_PVR3 | 7:0 | SDCLKFSEL[7:0] | | | | | | | |
| | | 15:8 | DRVSEL[1:0] | | | | | CLKGSEL | SDCLKFSEL[9:8] | |
| 0x68 | SDMMC_PVR4 | 7:0 | SDCLKFSEL[7:0] | | | | | | | |
| | | 15:8 | DRVSEL[1:0] | | | | | CLKGSEL | SDCLKFSEL[9:8] | |
| 0x6A | SDMMC_PVR5 | 7:0 | SDCLKFSEL[7:0] | | | | | | | |
| | | 15:8 | DRVSEL[1:0] | | | | | CLKGSEL | SDCLKFSEL[9:8] | |
| 0x6C | SDMMC_PVR6 | 7:0 | SDCLKFSEL[7:0] | | | | | | | |
| | | 15:8 | DRVSEL[1:0] | | | | | CLKGSEL | SDCLKFSEL[9:8] | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x6E | SDMMC_PVR7 | 7:0 | SDCLKFSEL[7:0] | | | | | | | | |
| | | 15:8 | DRVSEL[1:0] | | | | | CLKGSEL | SDCLKFSEL[9:8] | | |
| 0x70 ... 0xFB | Reserved | | | | | | | | | | |
| 0xFC | SDMMC_SISR | 7:0 | | | | | | | INTSSL[1:0] | | |
| | | 15:8 | | | | | | | | | |
| 0xFE | SDMMC_HCVR | 7:0 | SVER[7:0] | | | | | | | | |
| | | 15:8 | VVER[7:0] | | | | | | | | |
| 0x0100 ... 0x01FF | Reserved | | | | | | | | | | |
| 0x0200 | SDMMC_APSR | 7:0 | | | | | HDATLL[3:0] | | | | |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x0204 | SDMMC_MC1R | 7:0 | FCD | RSTN | BOOTA | OPD | DDR | | CMDTYP[1:0] | | |
| 0x0205 | SDMMC_MC2R | 7:0 | | | | | | | ABOOT | SRESP | |
| 0x0206 ... 0x0207 | Reserved | | | | | | | | | | |
| 0x0208 | SDMMC_ACR | 7:0 | | | | | | | BMAX[1:0] | | |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x020C | SDMMC_CC2R | 7:0 | | | | | | | | FSDCLKD | |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x0210 | SDMMC_RTC1R | 7:0 | | | | | | | | TMREN | |
| 0x0211 | SDMMC_RTC2R | 7:0 | | | | | | | | RLD | |
| 0x0212 ... 0x0213 | Reserved | | | | | | | | | | |
| 0x0214 | SDMMC_RTCVR | 7:0 | | | | | TCVAL[3:0] | | | | |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x0218 | SDMMC_RTISTER | 7:0 | | | | | | | | TEVT | |
| 0x0219 | SDMMC_RTISIER | 7:0 | | | | | | | | TEVT | |
| 0x021A ... 0x021B | Reserved | | | | | | | | | | |
| 0x021C | SDMMC_RTISTR | 7:0 | | | | | | | | TEVT | |
| 0x021D | SDMMC_RTSSR | 7:0 | | | | | | | TEVTSLOT[1:0] | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x021E ... 0x021F | Reserved | | | | | | | | | |
| 0x0220 | SDMMC_TUNCR | 7:0 | | | | | | | | SMPLPT |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0224 ... 0x022F | Reserved | | | | | | | | | |
| 0x0230 | SDMMC_CACR | 7:0 | | | | | | | | CAPWREN |
| | | 15:8 | KEY[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0234 ... 0x023F | Reserved | | | | | | | | | |
| 0x0240 | SDMMC_CALCR | 7:0 | | BPEN | TUNDIS | ALWYSON | CLKDIV[2:0] | | | EN |
| | | 15:8 | CNTVAL[7:0] | | | | | | | |
| | | 23:16 | CALNBP[3:0] | | | | CALN[3:0] | | | |
| | | 31:24 | CALPBP[3:0] | | | | CALP[3:0] | | | |

### 50.12.1 SDMMC SDMA System Address / Argument 2 Register

**Name:** SDMMC_SSAR
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** Read/Write

This register contains the physical system memory address used for SDMA transfers or the second argument for Auto CMD23.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR/ARG2[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR/ARG2[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR/ARG2[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR/ARG2[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR/ARG2[31:0]** SDMA System Address/Argument 2
ADDR: This field is the system memory address for a SDMA transfer. When the SDMMC stops an SDMA transfer, this field points to the system address of the next contiguous data position. This field can be accessed only if no transaction is executing (i.e., after a transaction has stopped). Read operations during transfers may return an invalid value. An interrupt can be generated to instruct the software to update this field. Writing the next system address of the next data position restarts the SDMA transfer.

ARG2: This field is used with Auto CMD23 to set a 32-bit block count value to the CMD23 argument while executing Auto CMD23. If Auto CMD23 is used with ADMA, the full 32-bit block count value can be used. If Auto CMD23 is used without ADMA, the available block count value is limited by SDMMC_BCR. In this case, 65535 blocks is the maximum value.

### 50.12.2 SDMMC Block Size Register

**Name:** SDMMC_BSR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | BOUNDARY[2:0] | | | | | BLKSIZE[9:8] | |
| Access | | R/W | R/W | R/W | | | R/W | R/W |
| Reset | | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BLKSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 14:12 – BOUNDARY[2:0]** SDMA Buffer Boundary
This field specifies the size of the contiguous buffer in the system memory. The SDMA transfer waits at every boundary specified by this field and the SDMMC generates the DMA Interrupt to instruct the software to update SDMMC_SSAR. If this field is set to 0 (buffer size = 4 Kbytes), the lowest 12 bits of SDMMC_SSAR.ADDRESS point to data in the contiguous buffer, and the upper 20 bits point to the location of the buffer in the system memory. This function is active when SDMMC_TMR.DMAEN is set.

| Value | Name | Description |
|---|---|---|
| 0 | 4K | 4-Kbyte boundary |
| 1 | 8K | 8-Kbyte boundary |
| 2 | 16K | 16-Kbyte boundary |
| 3 | 32K | 32-Kbyte boundary |
| 4 | 64K | 64-Kbyte boundary |
| 5 | 128K | 128-Kbyte boundary |
| 6 | 256k | 256-Kbyte boundary |
| 7 | 512K | 512-Kbyte boundary |

**Bits 9:0 – BLKSIZE[9:0]** Transfer Block Size
This field specifies the block size of data transfers for CMD14, CMD17, CMD18, CMD19, CMD24, CMD25, CMD53 and other data transfer commands such as CMD6, CMD8, ACMD13 and ACMD51. Values ranging from 1 to 512 can be set. It can be accessed only if no transaction is executing (i.e., after a transaction has stopped). Read operations during transfers may return an invalid value, and write operations are ignored.

### 50.12.3 SDMMC Block Count Register

**Name:** SDMMC_BCR
**Offset:** 0x06
**Reset:** 0x0000
**Property:** Read/Write

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BLKCNT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BLKCNT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – BLKCNT[15:0]** Block Count for Current Transfer
This field is used only if SDMMC_TMR.BCEN (Block Count Enable) is set to 1 and is valid only for multiple block transfers. BLKCNT is the number of blocks to be transferred and it must be set to a value between 1 and the maximum block count. The SDMMC decrements the block count after each block transfer and stops when the count reaches 0. When this field is set to 0, no data block is transferred.

This register should be accessed only when no transaction is executing (i.e., after transactions are stopped). During data transfer, read operations on this register may return an invalid value and write operations are ignored.

When a suspend command is completed, the number of blocks yet to be transferred can be determined by reading this register. Before issuing a resume command, the previously saved block count is restored.

### 50.12.4 SDMMC Argument 1 Register

**Name:** SDMMC_ARG1R
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ARG1[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ARG1[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ARG1[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ARG1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ARG1[31:0]** Argument 1
This register contains the SD command argument which is specified as the bit 39-8 of Command-Format in the “Physical Layer Simplified Specification V3.01” or “Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51” .

### 50.12.5 SDMMC Transfer Mode Register

**Name:** SDMMC_TMR
**Offset:** 0x0C
**Reset:** 0x0000
**Property:** Read/Write

This register is used to control data transfers. The user shall set this register before issuing a command which transfers data (see SDMMC_CR.DPSEL), or before issuing a Resume command. The user must save the value of this register when the data transfer is suspended (as a result of a Suspend command) and restore it before issuing a Resume command. To prevent data loss, this register cannot be written while data transactions are in progress. Writes to this register are ignored when SDMMC_PSR.CMDINHD is 1.

**Table 50-3. Determining the Transfer Type**

| MSBSEL | BCEN | BLKCNT (SDMMC_BCR) | Function |
|---|---|---|---|
| 0 | Don't care | Don't care | Single Transfer |
| 1 | 0 | Don't care | Infinite Transfer |
| 1 | 1 | Not Zero | Multiple Transfer |
| 1 | 1 | Zero | Stop Multiple Transfer |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | MSBSEL | DTDSEL | ACMDEN[1:0] | | BCEN | DMAEN |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 5 – MSBSEL** Multi/Single Block Selection
This bit is set to 1 when issuing multiple-block transfer commands using DAT line(s). For any other commands, set this bit to 0. If this bit is 0, it is not necessary to set SDMMC_BCR (see the table above, "Determining the Transfer Type").

**Bit 4 – DTDSEL** Data Transfer Direction Selection
This bit defines the direction of the DAT lines data transfers. Set this bit to 1 to transfer data from the device (SD Card/SDIO/e.MMC) to the SDMMC, and to 0 for all other commands.

0 (WRITE): Writes data from the SDMMC to the device.

1 (READ): Reads data from the device to the SDMMC.

**Bits 3:2 – ACMDEN[1:0]** Auto Command Enable
Two methods can be used to stop Multiple-block read and write operation:

- Auto CMD12: when the ACMDEN field is set to 1, the SDMMC issues CMD12 automatically when the last block transfer is completed. An Auto CMD12 error is indicated to SDMMC_ACESR. Auto CMD12 is not enabled if the command does not require CMD12.
- Auto CMD23: when the ACMDEN field is set to 2, the SDMMC issues a CMD23 automatically before issuing a command specified in SDMMC_CR.

The following conditions are required to use Auto CMD23:

- A memory card that supports CMD23 (SCR[33] = 1)
- If DMA is used, it must be ADMA (SDMA not supported).
- Only CMD18 or CMD25 is issued.

**Note:** The SDMMC does not check the command index.

Auto CMD23 can be used with or without ADMA. By writing SDMMC_CR, the SDMMC issues a CMD23 first and then issues a command specified by the SDMMC_CR.CMDIDX field. If CMD23 response errors are detected, the second command is not issued. A CMD23 error is indicated in SDMMC_ACESR. The CMD23 argument (32-bit block count value) is set in SDMMC_SSAR.

This field determines the use of auto command functions.

| Value | Name | Description |
|---|---|---|
| 0 | DISABLED | Auto Command Disabled |
| 1 | CMD12 | Auto CMD12 Enabled |
| 2 | CMD23 | Auto CMD23 Enabled |
| 3 | – | Reserved |

**Bit 1 – BCEN** Block Count Enable
This bit is used to enable SDMMC_BCR, which is only relevant for multiple block transfers. When this bit is 0, SDMMC_BCR is disabled, which is useful when executing an infinite transfer (see the table above). If an ADMA2 transfer is more than 65535 blocks, this bit is set to 0 and the data transfer length is designated by the Descriptor Table.

0 (DISABLED): Block count is disabled.

1 (ENABLED): Block count is enabled.

**Bit 0 – DMAEN** DMA Enable
This bit enables the DMA functionality described in section “Supporting DMA” in “SD Host Controller Simplified Specification V3.00” . DMA can be enabled only if it is supported as indicated by the bit SDMMC_CA0R.ADMA2SUP. One of the DMA modes can be selected using the field SDMMC_HC1R.DMASEL. If DMA is not supported, this bit is meaningless and then always reads 0. When this bit is set to 1, a DMA operation begins when the user writes to the upper byte of SDMMC_CR.

0 (DISABLED): DMA functionality is disabled.

1 (ENABLED): DMA functionality is enabled.

### 50.12.6 SDMMC Command Register

**Name:** SDMMC_CR
**Offset:** 0x0E
**Reset:** 0x0000
**Property:** Read/Write

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | CMDIDX[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMDTYP[1:0] | | DPSEL | CMDICEN | CMDCCEN | | RESPTYP[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | | 0 | 0 |

**Bits 13:8 – CMDIDX[5:0]** Command Index
This bit shall be set to the command number (CMD0–63, ACMD0–63) that is specified in bits 45–40 of the Command-Format in the “Physical Layer Simplified Specification V3.01” , “SDIO Simplified Specification V3.00” , and “Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51” .

**Bits 7:6 – CMDTYP[1:0]** Command Type

| Value | Name | Description |
|---|---|---|
| 0 | NORMAL | Other commands |
| 1 | SUSPEND | CMD52 to write “Bus Suspend” in the Card Common Control Registers (CCCR) (for SDIO only) |
| 2 | RESUME | CMD52 to write “Function Select” in the Card Common Control Registers (CCCR) (for SDIO only) |
| 3 | ABORT | CMD12, CMD52 to write “I/O Abort” in the Card Common Control Registers (CCCR) (for SDIO only) |

**Bit 5 – DPSEL** Data Present Select
This bit is set to 1 to indicate that data is present and shall be transferred using the DAT lines. It is set to 0 for the following:

- Commands using only CMD line (Ex. CMD52)
- Commands with no data transfer but using Busy signal on DAT[0] line (Ex. CMD38)
- Resume command

| Value | Description |
|---|---|
| 0 | No data present |
| 1 | Data present |

**Bit 4 – CMDICEN** Command Index Check Enable
If this bit is set to 1, the SDMMC checks the Index field in the response to see if it has the same value as the command index. If it has not, it is reported as a Command Index Error (CMDIDX) in SDMMC_EISTR. If this bit is set to 0, the Index field of the response is not checked.

0 (DISABLED): The Command Index Check is disabled.

1 (ENABLED): The Command Index Check is enabled.

**Bit 3 – CMDCCEN** Command CRC Check Enable
If this bit is set to 1, the SDMMC checks the CRC field in the response. If an error is detected, it is reported as a Command CRC Error (CMDCRC) in SDMMC_EISTR. If this bit is set to 0, the CRC field is not checked. The position of the CRC field is determined according to the length of the response.

0 (DISABLED): The Command CRC Check is disabled.

1 (ENABLED): The Command CRC Check is enabled.

**Bits 1:0 – RESPTYP[1:0]** Response Type
This field is set according to the response type expected for the command index (CMDIDX).

| Value | Name | Description |
|---|---|---|
| 0 | NORESP | No Response |
| 1 | RL136 | Response Length 136 |
| 2 | RL48 | Response Length 48 |
| 3 | RL48BUSY | Response Length 48 with Busy |

### 50.12.7 SDMMC Response Register x

**Name:** SDMMC_RRx
**Offset:** 0x10 + x*0x04 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CMDRESP[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CMDRESP[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMDRESP[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMDRESP[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CMDRESP[31:0]** Command Response
The table below describes the mapping of command responses from the SD_SDIO/e.MMC bus to these registers for each responses type. In this table, R[] refers to a bit range of the response data as transmitted on the SD_SDIO/e.MMC bus.

| Type of response | Meaning of response | Response field | Response register |
|---|---|---|---|
| R1, R1b (normal response) | Card Status | R[39:8] | SDMMC_RR0[31:0] |
| R1b (Auto CMD12 response) | Card Status for Auto CMD12 | R[39:8] | SDMMC_RR3[31:0] |
| R1 (Auto CMD23 response) | Card Status for Auto CMD23 | R[39:8] | SDMMC_RR3[31:0] |
| R2 (CID, CSD register) | CID or CSD register | R[127:8] | SDMMC_RR0[31:0]<br>SDMMC_RR1[31:0]<br>SDMMC_RR2[31:0]<br>SDMMC_RR3[23:0] |
| R3 (OCR register) | OCR register for memory | R[39:8] | SDMMC_RR0[31:0] |
| R4 (OCR register) | OCR register for I/O | R[39:8] | SDMMC_RR0[31:0] |
| R5, R5b | SDIO response | R[39:8] | SDMMC_RR0[31:0] |

.....continued

| Type of response | Meaning of response | Response field | Response register |
|---|---|---|---|
| R6 (Published RCA response) | New published RCA[31:16] and Card status bits | R[39:8] | SDMMC_RR0[31:0] |

### 50.12.8 SDMMC Buffer Data Port Register

**Name:** SDMMC_BDPR
**Offset:** 0x20
**Reset:** –
**Property:** Read/Write

**Note:** The reset value is an unpredictable value read from the dual port RAM.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BUFDATA[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BUFDATA[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BUFDATA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BUFDATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 31:0 – BUFDATA[31:0]** Buffer Data
The SDMMC data buffer can be accessed through this 32-bit Data Port register.

### 50.12.9 SDMMC Present State Register

**Name:** SDMMC_PSR
**Offset:** 0x24
**Reset:** 0x00F80000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CMDLL |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DATLL[3:0] | | | | WRPPL | CARDDPL | CARDSS | CARDINS |
| Access | R | R | R | R | R | R | R | R |
| Reset | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | BUFRDEN | BUFWREN | RTACT | WTACT |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | DLACT | CMDINHD | CMDINHC |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 24 – CMDLL** CMD Line Level
This status is used to check the CMD line level to recover from errors, and for debugging.

**Bits 23:20 – DATLL[3:0]** DAT[3:0] Line Level
This status is used to check the DAT line level to recover from errors, and for debugging. This is especially useful in detecting the Busy signal level from DAT[0].

**Bit 19 – WRPPL** Write Protect Pin Level
The Write Protect Switch is supported for memory and combo cards. This bit reflects the SDMMC_WP pin.

| Value | Description |
|---|---|
| 0 | Write protected (SDMMC_WP = 0) |
| 1 | Write enabled (SDMMC_WP = 1) |

**Bit 18 – CARDDPL** Card Detect Pin Level
This bit reflects the inverse value of the SDMMC_CD pin. Debouncing is not performed on this bit. This bit may be valid when CARDSS is set to 1, but it is not guaranteed because of the propagation delay. Use of this bit is limited to testing since it must be debounced by software.

| Value | Description |
|---|---|
| 0 | No card present (SDMMC_CD = 1). |

| Value | Description |
|---|---|
| 1 | Card present (SDMMC_CD = 0). |

**Bit 17 – CARDSS** Card State Stable
This bit is used for testing. If it is 0, the CARDDPL is not stable. If this bit is set to 1, it means that the CARDDPL is stable. No Card state can be detected if this bit is set to 1 and CARDINS is set to 0.

The Software Reset For All (SWRSTALL) in SDMMC_SRR does not affect this bit.

| Value | Description |
|---|---|
| 0 | Reset or debouncing. |
| 1 | No card or card inserted. |

**Bit 16 – CARDINS** Card Inserted
This bit indicates whether a card has been inserted. The SDMMC debounces this signal so that the user does not need to wait for it to stabilize.

A change from 0 to 1 raises the Card Insertion (CINS) status flag in SDMMC_NISTR if SDMMC_NISTER.CINS is set to 1. An interrupt is generated if SDMMC_NISIER.CINS is set to 1.

A change from 1 to 0 raises the Card Removal (CREM) status flag in SDMMC_NISTR if SDMMC_NISTER.CREM is set to 1. An interrupt is generated if SDMMC_NISIER.CREM is set to 1.

The Software Reset For All (SWRSTALL) in SDMMC_SRR does not affect this bit.

**Bit 11 – BUFRDEN** Buffer Read Enable
This bit is used for nonDMA read transfers. This flag indicates that valid data exists in the SDMMC data buffer. If this bit is 1, readable data exists in the buffer.

A change from 1 to 0 occurs when all the block data is read from the buffer.

A change from 0 to 1 occurs when block data is ready in the buffer. This raises the Buffer Read Ready (BRDRDY) status flag in SDMMC_NISTR if SDMMC_NISTER.BRDRDY is set to 1. An interrupt is generated if SDMMC_NISIER.BRDRDY is set to 1.

**Bit 10 – BUFWREN** Buffer Write Enable
This bit is used for nonDMA write transfers. This flag indicates if space is available for write data. If this bit is 1, data can be written to the buffer.

A change from 1 to 0 occurs when all the block data are written to the buffer.

A change from 0 to 1 occurs when top of block data can be written to the buffer. This raises the Buffer Write Ready (BWRRDY) status flag in SDMMC_NISTR if SDMMC_NISTER.BWRRDY is set to 1. An interrupt is generated if SDMMC_NISIER.BWRRDY is set to 1.

**Bit 9 – RTACT** Read Transfer Active
This bit is used to detect completion of a read transfer. See section “Read Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for more details on the sequence of events.

This bit is set to 1 in either of the following conditions:

- After the end bit of the read command.
- When a read operation is restarted by writing a 1 to SDMMC_BGCR.CONTR (Continue Request).

This bit is cleared to 0 in either of the following conditions:

- When the last data block as specified by Transfer Block Size (BLKSIZE) is transferred to the system.
- In case of ADMA2, end of read is designated by the descriptor table.
- When all valid data blocks in the SDMMC have been transferred to the system and no current block transfers are being sent as a result of the Stop At Block Gap Request (STPBGR) of SDMMC_BGCR being set to 1.

A change from 1 to 0 raises the Transfer Complete (TRFC) status flag in SDMMC_NISTR if SDMMC_NISTER.TRFC is set to 1. An interrupt is generated if SDMMC_NISIER.TRFC is set to 1.

**Bit 8 – WTACT** Write Transfer Active
This bit indicates a write transfer is active. If this bit is 0, it means no valid write data exists in the SDMMC. See section “Write Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for more details on the sequence of events.

This bit is set to 1 in either of the following conditions:

- After the end bit of the write command.
- When a write operation is restarted by writing a 1 to SDMMC_BGCR.CONTR (Continue Request).

This bit is cleared to 0 in either of the following conditions:

- After getting the CRC status of the last data block as specified by the transfer count (single and multiple). In case of ADMA2, transfer count is designated by the descriptor table.
- After getting the CRC status of any block where a data transmission is about to be stopped by a Stop At Block Gap Request (STPBGR) of SDMMC_BGCR.

During a write transaction and as the result of the Stop At Block Gap Request (STPBGR) being set, a change from 1 to 0 raises the Block Gap Event (BLKGE) status flag in SDMMC_NISTR if SDMMC_NISTER.BLKGE is set to 1. An interrupt is generated if BLKGE is set to 1 in SDMMC_NISIER. This status is useful to determine whether nonDAT line commands can be issued during Write Busy.

**Bit 2 – DLACT** DAT Line Active
This bit indicates whether one of the DAT lines on the bus is in use.

In the case of read transactions:

- This status indicates whether a read transfer is executing on the bus. A change from 1 to 0 resulting from setting the Stop At Block Gap Request (STPBGR) raises the Block Gap Event (BLKGE) status flag in SDMMC_NISTR if SDMMC_NISTER.BLKGE is set to 1. An interrupt is generated if SDMMC_NISIER.BLKGE is set to 1. See the section “Read Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for details on timing.
- This bit is set in either of the following cases:
  – After the end bit of the read command.
  – When writing 1 to SDMMC_BGCR.CONTR (Continue Request) to restart a read transfer.
- This bit is SDMMC cleared in either of the following cases:
  – When the end bit of the last data block is sent from the bus to the SDMMC. In case of ADMA2, the last block is designated by the last transfer of the Descriptor Table.
  – When a read transfer is stopped at the block gap initiated by a Stop At Block Gap Request (STPBGR).
- The SDMMC stops a read operation at the start of the interrupt cycle by driving the Read Wait (DAT[2] line) or by stopping the SD Clock. If the Read Wait signal is already driven (due to the fact

that the data buffer cannot receive data), the SDMMC can continue to stop the read operation by driving the Read Wait signal. It is necessary to support the Read Wait in order to use the Suspend/ Resume operation.

In the case of write transactions:

- This status indicates that a write transfer is executing on the bus. A change from 1 to 0 raises the Transfer Complete (TRFC) status flag in SDMMC_NISTR if SDMMC_NISTER.TRFC is set to 1. An interrupt is generated if SDMMC_NISIER.TRFC is set to 1. See the section “Write Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for details on timing.
- This bit is set in either of the following cases:
  – After the end bit of the write command.
  – When writing 1 to SDMMC_BGCR.CONTR (Continue Request) to continue a write transfer.
- This bit is cleared in either of the following cases:
  – When the card releases Write Busy of the last data block. If the card does not drive a Busy signal for 8 SDCLK, the SDMMC considers the card drive “Not Busy”. In the case of ADMA2, the last block is designated by the last transfer of the Descriptor Table.
  – When the card releases Write Busy prior to wait for write transfer as a result of a Stop At Block Gap Request (STPBGR).

Command with Busy:

This status indicates whether a command that indicates Busy (ex. erase command for memory) is executing on the bus. This bit is set to 1 after the end bit of the command with Busy and cleared when Busy is deasserted. A change from 1 to 0 raises the Transfer Complete (TRFC) status flag in SDMMC_NISTR if SDMMC_NISTER.TRFC is set to 1. An interrupt is generated if SDMMC_NISIER.TRFC is set to 1. See Figures 2.11 to 2.13 in the “SD Host Controller Simplified Specification V3.00” .

| Value | Description |
|---|---|
| 0 | DAT line inactive. |
| 1 | DAT line active. |

**Bit 1 – CMDINHD** Command Inhibit (DAT)
This status bit is 1 if either the DAT Line Active (DLACT) or the Read Transfer Active (RTACT) is set to 1. If this bit is 0, it indicates that the SDMMC can issue the next command. Commands with a Busy signal belong to Command Inhibit (DAT) (ex. R1b, R5b type). A change from 1 to 0 raises the Transfer Complete (TRFC) status flag in SDMMC_NISTR if SDMMC_NISTER.TRFC is set to 1. An interrupt is generated if SDMMC_NISIER.TRFC is set to 1.

Note: The software can save registers in the 000–00Dh range for a suspend transaction after this bit has changed from 1 to 0.

| Value | Description |
|---|---|
| 0 | Can issue a command which uses the DAT line(s). |
| 1 | Cannot issue a command which uses the DAT line(s). |

**Bit 0 – CMDINHC** Command Inhibit (CMD)
If this bit is 0, it indicates the CMD line is not in use and the SDMMC can issue a command using the CMD line. This bit is set to 1 immediately after SDMMC_CR is written. This bit is cleared when the

command response is received. Auto CMD12 and Auto CMD23 consist of two responses. In this case, this bit is not cleared by the CMD12 or CMD23 response, but by the Read/Write command response.

Status issuing Auto CMD12 is not read from this bit. So, if a command is issued during Auto CMD12 operation, the SDMMC manages to issue both commands: CMD12 and a command set by SDMMC_CR.

Even if the Command Inhibit (DAT) is set to 1, commands using only the CMD line can be issued if this bit is 0.

A change from 1 to 0 raises the Command Complete (CMDC) status flag in SDMMC_NISTR if SDMMC_NISTER.CMDC is set to 1. An interrupt is generated if SDMMC_NISIER.CMDC is set to 1.

If the SDMMC cannot issue the command because of a command conflict error (see SDMMC_EISTR.CMDCRC) or because of a ‘Command Not Issued By Auto CMD12’ error (see section “SDMMC Auto CMD Error Status Register”), this bit remains 1 and Command Complete is not set.

| Value | Description |
|---|---|
| 0 | Can issue a command using only CMD line. |
| 1 | Cannot issue a command. |

### 50.12.10 SDMMC Host Control 1 Register (SD_SDIO)

**Name:** SDMMC_HC1R (SD_SDIO)
**Offset:** 0x28
**Reset:** 0x00
**Property:** Read/Write

**Note:** This register configuration is specific to the SD/SDIO operation mode.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CARDDSEL | CARDDTL | | DMASEL[1:0] | | HSEN | DW | LEDCTRL |
| Access | R/W | R/W | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – CARDDSEL** Card Detect Signal Selection
This bit selects the source for the card detection.

| Value | Description |
|---|---|
| 0 | The SDMMC_CD pin is selected. |
| 1 | The Card Detect Test Level (CARDDTL) is selected (for test purposes). |

**Bit 6 – CARDDTL** Card Detect Test Level
This bit is enabled while the Card Detect Signal Selection (CARDDSEL) is set to 1 and it indicates whether the card is inserted or not.

| Value | Description |
|---|---|
| 0 | No card. |
| 1 | Card inserted. |

**Bits 4:3 – DMASEL[1:0]** DMA Select
One of the supported DMA modes can be selected. The DMA modes supported are given in SDMMC_CA0R. Use of a selected DMA is determined by DMA Enable (DMAEN) in SDMMC_TMR.

| Value | Name | Description |
|---|---|---|
| 0 | SDMA | SDMA is selected |
| 1 | – | Reserved |
| 2 | ADMA32 | 32-bit Address ADMA2 is selected |
| 3 | – | Reserved |

**Bit 2 – HSEN** High Speed Enable
Before setting this bit, the user must check High Speed Support (HSSUP) in SDMMC_CA0R.

If this bit is set to 0 (default), the SDMMC outputs CMD line and DAT lines at the falling edge of the SD clock (up to 25 MHz). If this bit is set to 1, the SDMMC outputs the CMD line and the DAT lines at the rising edge of the SD clock (up to 50 MHz).

If Preset Value Enable (PVALEN) in SDMMC_HC2R is set to 1, the user needs to reset SD Clock Enable (SDCLKEN) before changing this bit to avoid generating clock glitches. After setting this bit to 1, the user sets SDCLEN to 1 again.

**Note:** This bit is effective only if SDMMC_MC1R.DDR is set to 0.

**Note:** The clock divider (DIV) in SDMMC_CCR must be set to a value different from 0 when HSEN is 1.

| Value | Description |
|---|---|
| 0 | Normal Speed mode. |
| 1 | High Speed mode. |

**Bit 1 – DW** Data Width
This bit selects the data width of the SDMMC. It must be set to match the data width of the card.

0 (1_BIT): 1-bit mode.

1 (4_BIT): 4-bit mode.

**Note:** If the Extended Data Transfer Width is 1, this bit has no effect and the data width is 8-bit mode.

**Bit 0 – LEDCTRL** LED Control
This bit is used to caution the user not to remove the card while it is being accessed. If the software is going to issue multiple commands, this bit is set to 1 during all transactions.

0 (OFF): LED off.

1 (ON): LED on.

### 50.12.11 SDMMC Host Control 1 Register (e.MMC)

**Name:** SDMMC_HC1R (e.MMC)
**Offset:** 0x28
**Reset:** 0x00
**Property:** Read/Write

**Note:** This register configuration is specific to the e.MMC operation mode.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | EXTDW | DMASEL[1:0] | | HSEN | DW | |
| Access | | | R/W | R/W | R/W | R/W | R/W | |
| Reset | | | 0 | 0 | 0 | 0 | 0 | |

**Bit 5 – EXTDW** Extended Data Width
This bit controls the 8-bit Bus Width mode for embedded devices. Support of this function is indicated in 8-bit Support for Embedded Device in SDMMC_CA0R. If a device supports the 8-bit mode, this may be set to 1. If this bit is 0, the bus width is controlled by Data Width (DW).

**Bits 4:3 – DMASEL[1:0]** DMA Select
One of the supported DAM modes can be selected. The DMA modes supported are given in SDMMC_CA0R. Use of selected DMA is determined by DMA Enable (DMAEN) in SDMMC_TMR.

| Value | Name | Description |
|---|---|---|
| 0 | SDMA | SDMA is selected |
| 1 | – | Reserved |
| 2 | ADMA32 | 32-bit Address ADMA2 is selected |
| 3 | – | Reserved |

**Bit 2 – HSEN** High Speed Enable
Before setting this bit, the user must check High Speed Support (HSSUP) in SDMMC_CA0R.

If this bit is set to 0 (default), the SDMMC outputs CMD line and DAT lines at the falling edge of the SD clock (up to 25 MHz). If this bit is set to 1, the SDMMC outputs the CMD line and the DAT lines at the rising edge of the SD clock (up to 50 MHz).

If Preset Value Enable (PVALEN) in SDMMC_HC2R is set to 1, the user needs to reset the SD Clock Enable (SDCLKEN) before changing this bit to avoid generating clock glitches. After setting this bit to 1, the user sets SDCLEN to 1 again.

**Note:** This bit is effective only if SDMMC_MC1R.DDR is set to 0.

**Note:** The clock divider (DIV) in SDMMC_CCR must be set to a value different from 0 when HSEN is 1.

| Value | Description |
|---|---|
| 0 | Normal Speed mode. |
| 1 | High Speed mode. |

**Bit 1 – DW** Data Width
This bit selects the data width of the SDMMC. It must be set to match the data width of the card.

0 (1_BIT): 1-bit mode.

1 (4_BIT): 4-bit mode.

**Note:** If the Extended Data Transfer Width is 1, this bit has no effect and the data width is 8-bit mode.

### 50.12.12 SDMMC Power Control Register

**Name:** SDMMC_PCR
**Offset:** 0x29
**Reset:** 0x0E
**Property:** Read/Write

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SDBPWR |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – SDBPWR** SD Bus Power
This bit is automatically cleared by the SDMMC if the card is removed. If this bit is cleared, the SDMMC stops driving SDMMC_CMD and SDMMC_DAT[7:0] (tri-state) and drives SDMMC_CK to low level.

### 50.12.13 SDMMC Block Gap Control Register (SD_SDIO)

**Name:** SDMMC_BGCR (SD_SDIO)
**Offset:** 0x2A
**Reset:** 0x00
**Property:** Read/Write

**Note:** This register configuration is specific to the SD/SDIO operation mode.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | INTBG | RWCTRL | CONTR | STPBGR |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 3 – INTBG** Interrupt at Block Gap
This bit is valid only in 4-bit mode of the SDIO card and selects a sample point in the interrupt cycle. Setting to 1 enables interrupt detection at the block gap for a multiple block transfer. If the SDIO card cannot signal an interrupt during a multiple block transfer, this bit should be set to 0. When the software detects an SDIO card insertion, it sets this bit according to the CCCR of the SDIO card.

| Value | Name | Description |
|---|---|---|
| 0 | DISABLED | Interrupt detection disabled. |
| 1 | ENABLED | Interrupt detection enabled. |

**Bit 2 – RWCTRL** Read Wait Control
The Read Wait control is optional for SDIO cards. If the card supports Read Wait, set this bit to enable use of the Read Wait protocol to stop read data using the SDMMC_DAT[2] line. Otherwise, the SDMMC stops the SDCLK to hold read data, which restricts command generation. When the software detects an SD card insertion, this bit must be set according to the CCCR of the SDIO card. If the card does not support Read Wait, this bit shall never be set to 1, otherwise an SDMMC_DAT line conflict may occur. If this bit is set to 0, Suspend/Resume cannot be supported.

| Value | Description |
|---|---|
| 0 | Disables Read Wait control. |
| 1 | Enables Read Wait control. |

**Bit 1 – CONTR** Continue Request
This bit is used to restart a transaction which was stopped using a Stop At Block Gap Request (STPBGR). To cancel stop at the block gap, set STPBGR to 0 and set this bit to 1 to restart the transfer.

The SDMMC automatically clears this bit in either of the following cases:

– In the case of a read transaction, the DAT Line Active (DLACT) changes from 0 to 1 as a read transaction restarts.

– In the case of a write transaction, the Write Transfer Active (WTACT) changes from 0 to 1 as the write transaction restarts.

Therefore, it is not necessary to set this bit to 0. If STPBGR is set to 1, any write to this bit is ignored.

See the “Abort Transaction” and “Suspend/Resume” sections in the “SD Host Controller Simplified Specification V3.00” for more details.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Restart. |

**Bit 0 – STPBGR** Stop At Block Gap Request
This bit is used to stop executing read and write transactions at the next block gap for nonDMA, SDMA, and ADMA transfers. The user must leave this bit set to 1 until Transfer Complete (TRFC) in SDMMC_NISTR. Clearing both Stop At Block Gap Request and Continue Request does not cause the transaction to restart. This bit can be set whether the card supports the Read Wait signal or not.

During read transfers, the SDMMC stops the transaction by using the Read Wait signal (SDMMC_DAT[2]) if supported, or by stopping the SD clock otherwise.

In case of write transfers in which the user writes data to SDMMC_BDPR, this bit must be set to 1 after all the block of data is written. If this bit is set to 1, the user does not write data to SDMMC_BDPR.

This bit affects Read Transfer Active (RTACT), Write Transfer Active (WTACT), DAT Line Active (DLACT) and Command Inhibit (DAT) (CMDINHD) in SDMMC_PSR.

See the “Abort Transaction” and “Suspend/Resume” sections in the “SD Host Controller Simplified Specification V3.00” for more details.

| Value | Description |
|---|---|
| 0 | Transfer |
| 1 | Stop |

### 50.12.14 SDMMC Block Gap Control Register (e.MMC)

**Name:** SDMMC_BGCR (e.MMC)
**Offset:** 0x2A
**Reset:** 0x00
**Property:** Read/Write

**Note:** This register configuration is specific to the e.MMC operation mode.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CONTR | STPBGR |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – CONTR** Continue Request
This bit is used to restart a transaction which was stopped using a Stop At Block Gap Request (STPBGR). To cancel stop at the block gap, set STPBGR to 0 and set this bit to 1 to restart the transfer.

The SDMMC automatically clears this bit in either of the following cases:

– In the case of a read transaction, the DAT Line Active (DLACT) changes from 0 to 1 as a read transaction restarts.

– In the case of a write transaction, the Write Transfer Active (WTACT) changes from 0 to 1 as the write transaction restarts.

Therefore, it is not necessary to set this bit to 0. If STPBGR is set to 1, any write to this bit is ignored.

See the “Abort Transaction” and “Suspend/Resume” sections in the “SD Host Controller Simplified Specification V3.00” for more details.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Restart. |

**Bit 0 – STPBGR** Stop At Block Gap Request
This bit is used to stop executing read and write transactions at the next block gap for nonDMA, SDMA, and ADMA transfers. The user must leave this bit set to 1 until Transfer Complete (TRFC) in SDMMC_NISTR. Clearing both Stop At Block Gap Request and Continue Request does not cause the transaction to restart. This bit can be set whether the card supports the Read Wait signal or not.

During read transfers, the SDMMC stops the transaction by using the Read Wait signal (SDMMC_DAT[2]) if supported, or by stopping the SD clock otherwise.

In case of write transfers in which the user writes data to SDMMC_BDPR, this bit must be set to 1 after all the block of data is written. If this bit is set to 1, the user does not write data to SDMMC_BDPR.

This bit affects Read Transfer Active (RTACT), Write Transfer Active (WTACT), DAT Line Active (DLACT) and Command Inhibit (DAT) (CMDINHD) in SDMMC_PSR.

See the “Abort Transaction” and “Suspend/Resume” sections in the “SD Host Controller Simplified Specification V3.00” for more details.

| Value | Description |
|---|---|
| 0 | Transfer |

| Value | Description |
|---|---|
| 1 | Stop |

### 50.12.15 SDMMC Wakeup Control Register (SD_SDIO)

**Name:** SDMMC_WCR (SD_SDIO)
**Offset:** 0x2B
**Reset:** 0x00
**Property:** Read/Write

**Note:** This register configuration is specific to the SD/SDIO operation mode.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | WKENCREM | WKENCINS | WKENCINT |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – WKENCREM** Wakeup Event Enable on Card Removal
This bit enables a wakeup event via Card Removal (CREM) in SDMMC_NISTR. FN_WUS (Wakeup Support) in the CIS (Card Information Structure) does not affect this bit.

0 (DISABLED): Wakeup Event disabled.

1 (ENABLED): Wakeup Event enabled.

**Bit 1 – WKENCINS** Wakeup Event Enable on Card Insertion
This bit enables a wakeup event via Card Insertion (CINS) in SDMMC_NISTR. FN_WUS (Wakeup Support) in the CIS (Card Information Structure) does not affect this bit.

0 (DISABLED): Wakeup Event disabled.

1 (ENABLED): Wakeup Event enabled.

**Bit 0 – WKENCINT** Wakeup Event Enable on Card Interrupt
This bit enables a wakeup event via Card Interrupt (CINT) in SDMMC_NISTR. This bit can be set to 1 if FN_WUS (Wakeup Support) in the CIS (Card Information Structure) is set to 1 in the SDIO card.

0 (DISABLED): Wakeup Event disabled.

1 (ENABLED): Wakeup Event enabled.

### 50.12.16 SDMMC Clock Control Register

**Name:** SDMMC_CCR
**Offset:** 0x2C
**Reset:** 0x0000
**Property:** Read/Write

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SDCLKFSEL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | USDCLKFSEL[1:0] | | CLKGSEL | | | SDCLKEN | INTCLKS | INTCLKEN |
| Access | R/W | R/W | R/W | | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | | | 0 | 0 | 0 |

**Bits 15:8 – SDCLKFSEL[7:0]** SDCLK Frequency Select
This register is used to select the frequency of the SDCLK pin. There are two SDCLK Frequency modes according to Clock Generator Select (CLKGSEL).

The length of the clock divider (DIV) is extended to 10 bits (DIV[9:8] = USDCLKFSEL, DIV[7:0] = SDCLKFSEL)

– 10-bit Divided Clock Mode (CLKGSEL = 0): $f_{\mathrm{SDCLK}} = f_{\mathrm{BASECLK}}/(2 \times \mathrm{DIV})$. If DIV = 0 then $f_{\mathrm{SDCLK}} = f_{\mathrm{BASECLK}}$

– Programmable Clock Mode (CLKGSEL = 1): $f_{\mathrm{SDCLK}} = f_{\mathrm{MULTCLK}}/(\mathrm{DIV} + 1)$

When HSEN is set in SDMMC_HC1R, or DDR is set in SDMMC_MC1R, the clock divider (DIV) must be non-zero.This field depends on the setting of Preset Value Enable (PVALEN) in SDMMC_HC2R.

If PVALEN = 0, this field is set by the user.

If PVALEN = 1, this field is automatically set to a value specified in one of the SDMMC_PVR.

**Bits 7:6 – USDCLKFSEL[1:0]** Upper Bits of SDCLK Frequency Select
These bits expand the SDCLK Frequency Select (SDCLKFSEL) to 10 bits. These two bits are assigned to bit 09-08 of the clock divider as described in SDCLKFSEL.

**Bit 5 – CLKGSEL** Clock Generator Select
This bit is used to select the clock generator mode in the SDCLK Frequency Select field. If the Programmable mode is not supported (SDMMC_CA1R.CLKMULT (Clock Multiplier) set to 0), then this bit cannot be written and is always read at 0.

This bit depends on the setting of Preset Value Enable (PVALEN) in SDMMC_HC2R.

If PVALEN = 0, this bit is set by the user.

If PVALEN = 1, this bit is automatically set to a value specified in one of the SDMMC_PVRx.

| Value | Description |
|---|---|
| 0 | Divided Clock mode (BASECLK is used to generate SDCLK). |

| Value | Description |
|---|---|
| 1 | Programmable Clock mode (MULTCLK is used to generate SDCLK). |

**Bit 2 – SDCLKEN** SD Clock Enable
The SDMMC stops the SD Clock when writing this bit to 0. SDCLK Frequency Select (SDCLKFSEL) can be changed when this bit is 0. Then, the SDMMC maintains the same clock frequency until SDCLK is stopped (Stop at SDCLK = 0). If Card Inserted (CARDINS) in SDMMC_PSR is cleared, this bit is also cleared.

| Value | Description |
|---|---|
| 0 | SD Clock disabled |
| 1 | SD Clock enabled |

**Bit 1 – INTCLKS** Internal Clock Stable
This bit is set to 1 when the SD clock is stable after setting SDMMC_CCR.INTCLKEN (Internal Clock Enable) to 1. The user must wait to set SD Clock Enable (SDCLKEN) until this bit is set to 1.

| Value | Description |
|---|---|
| 0 | Internal clock not ready. |
| 1 | Internal clock ready. |

**Bit 0 – INTCLKEN** Internal Clock Enable
This bit is set to 0 when the SDMMC is not used or is awaiting a wakeup interrupt. In this case, its internal clock is stopped to reach a very low power state. Registers are still able to be read and written. The clock starts to oscillate when this bit is set to 1. Once the clock oscillation is stable, the SDMMC sets Internal Clock Stable (INTCLKS) in this register to 1.

This bit does not affect card detection.

| Value | Description |
|---|---|
| 0 | The internal clock stops. |
| 1 | The internal clock oscillates. |

### 50.12.17 SDMMC Timeout Control Register

**Name:** SDMMC_TCR
**Offset:** 0x2E
**Reset:** 0x00
**Property:** Read/Write

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | DTCVAL[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 3:0 – DTCVAL[3:0]** Data Timeout Counter Value
This value determines the interval at which DAT line timeouts are detected. For more information about timeout generation, see Data Timeout Error (DATTEO) in SDMMC_EISTR. When setting this register, the user can prevent inadvertent timeout events by clearing the Data Timeout Error Status Enable (in SDMMC_EISTER).

$$\mathrm{TIMEOUT}_{(\mu s)} = \frac{2^{13 + \mathrm{DTCVAL}}}{f_{\mathrm{FTEOCLK(MHz)}}}$$

**Note:** DTCVAL = $f_{(Hexa)}$ is reserved.

### 50.12.18 SDMMC Software Reset Register

**Name:** SDMMC_SRR
**Offset:** 0x2F
**Reset:** 0x00
**Property:** Read/Write

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SWRSTDAT | SWRSTCMD | SWRSTALL |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – SWRSTDAT** Software Reset for DAT Line
Only part of a data circuit is reset. The DMA circuit is also reset.

The following registers and bits are cleared by this bit:

"SDMMC Buffer Data Port Register"

– Buffer is cleared and initialized.

"SDMMC Present State Register"

– Buffer Read Enable (BUFRDEN)

– Buffer Write Enable (BUFWREN)

– Read Transfer Active (RTACT)

– Write Transfer Active (WTACT)

– DAT Line Active (DATLL)

– Command Inhibit (DAT) (CMDINHD)

"SDMMC Block Gap Control Register (SD_SDIO)"

– Continue Request (CONTR)

– Stop At Block Gap Request (STPBGR)

"SDMMC Normal Interrupt Status Register (SD_SDIO)"

– Buffer Read Ready (BRDRDY)

– Buffer Write Ready (BWRRDY)

– DMA Interrupt (DMAINT)

– Block Gap Event (BLKGE)

– Transfer Complete (TRFC)

| Value | Description |
|---|---|
| 0 | Work |
| 1 | Reset |

**Bit 1 – SWRSTCMD** Software Reset for CMD Line
Only part of a command circuit is reset.

The following registers and bits are cleared by this bit:

"SDMMC Present State Register"

– Command Inhibit (CMD) (CMDINHC)

"SDMMC Normal Interrupt Status Register (SD_SDIO)" and "SDMMC Normal Interrupt Status Register (e.MMC)"

– Command Complete (CMDC)

| Value | Description |
|---|---|
| 0 | Work |
| 1 | Reset |

**Bit 0 – SWRSTALL** Software Reset for All
This reset affects the entire SDMMC except the card detection circuit. During initialization, the SDMMC must be reset by setting this bit to '1'. This bit is automatically cleared to '0' when SDMMC_CA0R and SDMMC_CA1R are valid and the user can read them. If this bit is set to '1', the user should issue a reset command and reinitialize the card.

List of registers cleared to '0':

– "SDMMC SDMA System Address / Argument 2 Register"

– "SDMMC Block Size Register"

– "SDMMC Block Count Register"

– "SDMMC Argument 1 Register"

– "SDMMC Command Register"

– "SDMMC Transfer Mode Register"

– "SDMMC Response Register"

– "SDMMC Buffer Data Port Register"

– "SDMMC Present State Register" (except CMDLL, DATLL, WRPPL, CARDDDPL, CARDSS, CARDINS)

– "SDMMC Host Control 1 Register (SD_SDIO)"

– "SDMMC Host Control 1 Register (e.MMC)"

– "SDMMC Power Control Register"

– "SDMMC Block Gap Control Register (SD_SDIO)"

– "SDMMC Block Gap Control Register (e.MMC)"

– "SDMMC Wakeup Control Register (SD_SDIO)"

– "SDMMC Clock Control Register"

– "SDMMC Timeout Control Register"

– "SDMMC Normal Interrupt Status Register (SD_SDIO)"

– "SDMMC Error Interrupt Status Register (SD_SDIO)"

– "SDMMC Normal Interrupt Status Enable Register (SD_SDIO)"

– "SDMMC Error Interrupt Status Enable Register (SD_SDIO)"

– “SDMMC Normal Interrupt Signal Enable Register (SD_SDIO)”
– “SDMMC Error Interrupt Signal Enable Register (SD_SDIO)”
– “SDMMC Auto CMD Error Status Register”
– “SDMMC Host Control 2 Register (SD_SDIO)”
– “SDMMC ADMA Error Status Register”
– “SDMMC ADMA System Address Register”
– “SDMMC Slot Interrupt Status Register”
– “SDMMC e.MMC Control 1 Register”
– “SDMMC e.MMC Control 2 Register”
– “SDMMC AHB Control Register”
– “SDMMC Clock Control 2 Register”
– “SDMMC Retuning Control 1 Register”
– “SDMMC Retuning Counter Value Register”
– “SDMMC Retuning Interrupt Status Enable Register”
– “SDMMC Retuning Interrupt Signal Enable Register”
– “SDMMC Retuning Interrupt Status Register”
– “SDMMC Tuning Control Register”
– “SDMMC Capabilities Control Register” (except KEY)
– “SDMMC Calibration Control Register” (except CALN, CALP)

| Value | Description |
|---|---|
| 0 | Work |
| 1 | Reset |

### 50.12.19 SDMMC Normal Interrupt Status Register (SD_SDIO)

**Name:** SDMMC_NISTR (SD_SDIO)
**Offset:** 0x30
**Reset:** 0x0000
**Property:** Read/Write

**Note:** This register configuration is specific to the SD/SDIO operation mode.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ERRINT | | | | | | | CINT |
| Access | R | | | | | | | R/W |
| Reset | 0 | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CREM | CINS | BRDRDY | BWRRDY | DMAINT | BLKGE | TRFC | CMDC |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 15 – ERRINT** Error Interrupt
If any of the bits in SDMMC_EISTR are set, then this bit is set. Therefore, the user can efficiently test for an error by checking this bit first. This bit is read-only.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 8 – CINT** Card Interrupt
Writing this bit to '1' does not clear this bit. It is cleared by resetting the SD card interrupt factor. In 1-bit mode, the SDMMC detects the Card Interrupt without SDCLK to support wakeup. In 4-bit mode, the Card Interrupt signal is sampled during the interrupt cycle, so there are some sample delays between the interrupt signal from the SD card and the interrupt to the system.

When this bit has been set to '1' and the user needs to start this interrupt service, Card Interrupt Status Enable (CINT) in SDMMC_NISTER may be set to '0' in order to clear the card interrupt statuses latched in the SDMMC and to stop driving the interrupt signal to the system. After completion of the card interrupt service (it should reset interrupt factors in the SD card and the interrupt signal may not be asserted), set SDMMC_NISTER.CINT to '1' and start sampling the interrupt signal again.

Interrupt detected by DAT[1] is supported when there is one card per slot.

This bit can only be set to '1' if SDMMC_NISTER.CREM is set to '1'. An interrupt can only be generated if SDMMC_NISIER.CREM is set to '1'.

| Value | Description |
|---|---|
| 0 | No card interrupt. |
| 1 | Card interrupt. |

**Bit 7 – CREM** Card Removal
This status is set to '1' if Card Inserted (CARDINS) in SDMMC_PSR changes from '1' to '0'. When the user writes this bit to '1' to clear this status, the status of SDMMC_PSR.CARDINS must be confirmed

because the card detect state may possibly be changed when the user clears this bit and no interrupt event can be generated.

This bit can only be set to '1' if SDMMC_NISTER.CREM is set to '1'. An interrupt can only be generated if SDMMC_NISIER.CREM is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | Card state unstable or card inserted. |
| 1 | Card removed. |

**Bit 6 – CINS** Card Insertion
This status is set if Card Inserted (CARDINS) in SDMMC_PSR changes from '0' to '1'. When the user writes this bit to '1' to clear this status, the status of SDMMC_PSR.CARDINS must be confirmed because the card detect state may possibly be changed when the user clears this bit and no interrupt event can be generated.

This bit can only be set to '1' if SDMMC_NISTER.CINS is set to '1'. An interrupt can only be generated if SDMMC_NISIER.CINS is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | Card state unstable or card removed. |
| 1 | Card inserted. |

**Bit 5 – BRDRDY** Buffer Read Ready
This status is set to '1' if the Buffer Read Enable (BUFRDEN) changes from '0' to '1'. See BUFRDEN in SDMMC_PSR. While processing the tuning procedure (Execute Tuning (EXTUN) in SDMMC_HC2R is set to '1'), BRDRDY is set to '1' for every CMD19 execution.

This bit can only be set to '1' if SDMMC_NISTER.BRDRDY is set to '1'. An interrupt can only be generated if SDMMC_NISIER.BRDRDY is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | Not ready to read buffer. |
| 1 | Ready to read buffer. |

**Bit 4 – BWRRDY** Buffer Write Ready
This status is set to '1' if the Buffer Write Enable (BUFWREN) changes from '0' to '1'. See BUFWREN in SDMMC_PSR.

This bit can only be set to '1' if SDMMC_NISTER.BWRRDY is set to '1'. An interrupt can only be generated if SDMMC_NISIER.BWRRDY is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | Not ready to write buffer. |
| 1 | Ready to write buffer. |

**Bit 3 – DMAINT** DMA Interrupt
This status is set if the SDMMC detects the Host SDMA Buffer boundary during transfer. See SDMA Buffer Boundary (BOUNDARY) in SDMMC_BSR.

In case of ADMA, by setting the “int” field in the descriptor table, the SDMMC raises this status flag when the descriptor line is completed. This status flag does not rise after Transfer Complete (TRFC).

This bit can only be set to '1' if SDMMC_NISTER.DMAINT is set to '1'. An interrupt can only be generated if SDMMC_NISIER.DMAINT is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No DMA Interrupt. |
| 1 | DMA Interrupt. |

**Bit 2 – BLKGE** Block Gap Event
If the Stop At Block Gap Request (STPBGR) in SDMMC_BGCR is set to 1, this bit is set when either a read or a write transaction is stopped at a block gap. If STPBGR is not set to 1, this bit is not set to 1.

**In the case of a Read transaction:**

This bit is set at the falling edge of the DAT Line Active (DLACT) status (when the transaction is stopped at SD bus timing). The Read Wait must be supported in order to use this function. See section “Read Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” about the detailed timing.

**In the case of a Write transaction:**

This bit is set at the falling edge of the Write Transfer Active (WTACT) status (after getting the CRC status at SD bus timing). See section “Write Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for more details on the sequence of events.

This bit can only be set to '1' if SDMMC_NISTER.BLKGE is set to '1'. An interrupt can only be generated if SDMMC_NISIER.BLKGE is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No block gap event. |
| 1 | Transaction stopped at block gap. |

**Bit 1 – TRFC** Transfer Complete
This bit is set when a read/write transfer and a command with Busy is completed.

**In the case of a Read Transaction:**

This bit is set at the falling edge of the Read Transfer Active Status. The interrupt is generated in two cases. The first is when a data transfer is completed as specified by the data length (after the last data has been read to the system). The second is when data has stopped at the block gap and completed the data transfer by setting the Stop At Block Gap Request (STPBGR) in SDMMC_BGCR (after valid data has been read to the system). See section “Read Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for more details on the sequence of events.

**In the case of a Write Transaction:**

This bit is set at the falling edge of the DAT Line Active (DLACT) status. This interrupt is generated in two cases. The first is when the last data is written to the card as specified by the data length and the Busy signal is released. The second is when data transfers are stopped at the block gap by setting Stop At Block Gap Request (STPBGR) in SDMMC_BGCR and data transfers are completed. (After valid data is written to the card and the Busy signal is released). See section “Write Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for more details on the sequence of events.

**In the case of command with Busy:**

This bit is set when Busy is deasserted. See DAT Line Active (DLACT) and Command Inhibit (DAT) (CMDINHD) in SDMMC_PSR.

This bit can only be set to '1' if SDMMC_NISTER.TRFC is set to '1'. An interrupt can only be generated if SDMMC_NISIER.TRFC is set to '1'.

Writing this bit to '1' clears this bit.

The table below shows that Transfer Complete (TRFC) has a higher priority than Data Timeout Error (DATTEO). If both bits are set to '1', execution of a command can be considered to be completed.

| TRFC | DATTEO | Meaning of the status |
|---|---|---|
| 0 | 0 | Interrupted by another factor |
| 0 | 1 | Timeout occurred during transfer |
| 1 | Don’t Care | Command execution complete |

| Value | Description |
|---|---|
| 0 | Command execution is not complete. |
| 1 | Command execution is complete. |

**Bit 0 – CMDC** Command Complete
This bit is set when getting the end bit of the command response. Auto CMD12 and Auto CMD23 consist of two responses. Command Complete is not generated by the response of CMD12 or CMD23, but it is generated by the response of a read/write command. See Command Inhibit (CMD) in SDMMC_PSR for details on how to control this bit.

This bit can only be set to 1 if SDMMC_NISTER.CMDC is set to 1. An interrupt can only be generated if SDMMC_NISIER.CMDC is set to 1.

Writing this bit to 1 clears this bit.

The table below shows that Command Timeout Error (CMDTEO) has a higher priority than Command Complete (CMDC). If both bits are set to 1, it can be considered that the response was not received correctly.

| CMDC | CMDTEO | Meaning of the status |
|---|---|---|
| 0 | 0 | Interrupted by another factor |
| Don’t care | 1 | Response not received within 64 SDCLK cycles |
| 1 | 0 | Response received |

| Value | Description |
|---|---|
| 0 | No command complete. |
| 1 | Command complete. |

### 50.12.20 SDMMC Normal Interrupt Status Register (e.MMC)

**Name:** SDMMC_NISTR (e.MMC)
**Offset:** 0x30
**Reset:** 0x0000
**Property:** Read/Write

**Note:** This register configuration is specific to the e.MMC operation mode.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ERRINT | BOOTAR | | | | | | |
| Access | R | R/W | | | | | | |
| Reset | 0 | 0 | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | BRDRDY | BWRRDY | DMAINT | BLKGE | TRFC | CMDC |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 15 – ERRINT** Error Interrupt
If any of the bits in SDMMC_EISTR are set, then this bit is set. Therefore, the user can efficiently test for an error by checking this bit first. This bit is read-only.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 14 – BOOTAR** Boot Acknowledge Received
This bit is set to '1' when the SDMMC received a Boot Acknowledge pattern from the e.MMC.

This bit can only be set to '1' if SDMMC_NISTER.BOOTAR is set to '1'. An interrupt can only be generated if SDMMC_NISIER.BOOTAR is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | Boot Acknowledge pattern not received. |
| 1 | Boot Acknowledge pattern received. |

**Bit 5 – BRDRDY** Buffer Read Ready
This status is set to '1' if Buffer Read Enable (BUFRDEN) changes from '0' to '1'. See Buffer Read Enable (BUFRDEN) in SDMMC_PSR. While processing the tuning procedure (Execute Tuning (EXTUN) in SDMMC_HC2R is set to '1'), BRDRDY is set to '1' for every CMD19 execution.

This bit can only be set to '1' if SDMMC_NISTER.BRDRDY is set to '1'. An interrupt can only be generated if SDMMC_NISIER.BRDRDY is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | Not ready to read buffer. |
| 1 | Ready to read buffer. |

**Bit 4 – BWRRDY** Buffer Write Ready
This status is set to 1 if Buffer Write Enable (BUFWREN) changes from '0' to '1'. See Buffer Write Enable (BUFWREN) in SDMMC_PSR.

This bit can only be set to '1' if SDMMC_NISTER.BWRRDY is set to '1'. An interrupt can only be generated if SDMMC_NISIER.BWRRDY is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | Not ready to write buffer. |
| 1 | Ready to write buffer. |

**Bit 3 – DMAINT** DMA Interrupt
This status is set if the SDMMC detects the Host SDMA Buffer boundary during transfer. See SDMA Buffer Boundary (BOUNDARY) in SDMMC_BSR.

In case of ADMA, by setting “int” field in the descriptor table, the SDMMC raises this status flag when the descriptor line is completed. This status flag does not rise after the Transfer Complete (TRFC).

This bit can only be set to '1' if SDMMC_NISTER.DMAINT is set to '1'. An interrupt can only be generated if SDMMC_NISIER.DMAINT is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No DMA interrupt. |
| 1 | DMA interrupt. |

**Bit 2 – BLKGE** Block Gap Event
If the Stop At Block Gap Request (STPBGR) in SDMMC_BGCR is set to '1', this bit is set when either a read or a write transaction is stopped at a block gap. If STPBGR is not set to '1', this bit is not set to '1'.

**In the case of a Read transaction:**

This bit is set at the falling edge of the DAT Line Active (DLACT) status (when the transaction is stopped at SD bus timing). The Read Wait must be supported in order to use this function. See section “Read Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” about the detailed timing.

**In the case of a Write transaction:**

This bit is set at the falling edge of the Write Transfer Active (WTACT) status (after getting the CRC status at SD bus timing). See section “Write Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for more details on the sequence of events.

This bit can only be set to '1' if SDMMC_NISTER.BLKGE is set to '1'. An interrupt can only be generated if SDMMC_NISIER.BLKGE is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No block gap event. |
| 1 | Transaction stopped at block gap. |

**Bit 1 – TRFC** Transfer Complete
This bit is set when a read/write transfer and a command with Busy is completed.

**In the case of a Read Transaction:**

This bit is set at the falling edge of the Read Transfer Active Status. The interrupt is generated in two cases. The first is when a data transfer is completed as specified by the data length (after the last data has been read to the system). The second is when data has stopped at the block gap and completed the data transfer by setting the Stop At Block Gap Request (STPBGR) in SDMMC_BGCR (after valid data has been read to the system). See section “Read Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for more details on the sequence of events.

**In the case of a Write Transaction:**

This bit is set at the falling edge of the DAT Line Active (DLACT) status. This interrupt is generated in two cases. The first is when the last data is written to the card as specified by the data length and the Busy signal is released. The second is when data transfers are stopped at the block gap by setting Stop At Block Gap Request (STPBGR) in SDMMC_BGCR and data transfers are completed. (After valid data is written to the card and the Busy signal is released). See section “Write Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for more details on the sequence of events.

**In the case of command with Busy:**

This bit is set when Busy is deasserted. See DAT Line Active (DLACT) and Command Inhibit (DAT) (CMDINHD) in SDMMC_PSR.

This bit can only be set to '1' if SDMMC_NISTER.TRFC is set to '1'. An interrupt can only be generated if SDMMC_NISIER.TRFC is set to '1'.

Writing this bit to '1' clears this bit.

The table below shows that Transfer Complete (TRFC) has a higher priority than Data Timeout Error (DATTEO). If both bits are set to '1', execution of a command can be considered to be completed.

| TRFC | DATTEO | Meaning of the status |
|---|---|---|
| 0 | 0 | Interrupted by another factor |
| 0 | 1 | Timeout occurred during transfer |
| 1 | Don’t Care | Command execution complete |

| Value | Description |
|---|---|
| 0 | Command execution is not complete. |
| 1 | Command execution is complete. |

**Bit 0 – CMDC** Command Complete
This bit is set when getting the end bit of the command response. Auto CMD12 and Auto CMD23 consist of two responses. Command Complete is not generated by the response of CMD12 or CMD23, but it is generated by the response of a read/write command. See CMRINHC in SDMMC_PSR for details on how to control this bit.

This bit can only be set to '1' if SDMMC_NISTER.CMDC is set to '1'. An interrupt can only be generated if SDMMC_NISIER.CMDC is set to '1'.

Writing this bit to '1' clears this bit.

The table below shows that Command Timeout Error (CMDTEO) has a higher priority than Command Complete (CMDC). If both bits are set to '1', it can be considered that the response was not received correctly.

| CMDC | CMDTEO | Meaning of the status |
|---|---|---|
| 0 | 0 | Interrupted by another factor |
| Don’t care | 1 | Response not received within 64 SDCLK cycles |
| 1 | 0 | Response received |

| Value | Description |
|---|---|
| 0 | No command complete. |
| 1 | Command complete. |

### 50.12.21 SDMMC Error Interrupt Status Register (SD_SDIO)

**Name:** SDMMC_EISTR (SD_SDIO)
**Offset:** 0x32
**Reset:** 0x0000
**Property:** Read/Write

**Note:** This register configuration is specific to the SD/SDIO operation mode.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ADMA | ACMD |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CURLIM | DATEND | DATCRC | DATTEO | CMDIDX | CMDEND | CMDCRC | CMDTEO |
| Access | R/W | R/W | R/W | | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 9 – ADMA** ADMA Error
This bit is set to '1' when the SDMMC detects errors during an ADMA-based data transfer. The state of the ADMA at an error occurrence is saved in SDMMC_AESR.

In addition, the SDMMC raises this status flag when it detects some invalid description data (Valid = 0) at the ST_FDS state (see section “Advanced DMA” in the “SD Host Controller Simplified Specification V3.00”). ADMA Error Status (ERRST) in SDMMC_AESR indicates that an error occurred in ST_FDS state. The user may find that the Valid bit is not set at the error descriptor.

This bit can only be set to '1' if SDMMC_EISTER.ADMA is set to '1'. An interrupt can only be generated if SDMMC_EISIER.ADMA is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 8 – ACMD** Auto CMD Error
Auto CMD12 and Auto CMD23 use this error status. This bit is set to '1' when detecting that one of the 0 to 4 bits in SDMMC_ACESR[4:0] has changed from '0' to '1'. In the case of Auto CMD12, this bit is set to '1', not only when errors occur in Auto CMD12 but also when auto CMD12 is not executed due to the previous command error.

This bit can only be set to '1' if SDMMC_EISTER.ACMD is set to '1'. An interrupt can only be generated if SDMMC_EISIER.ACMD is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 7 – CURLIM** Current Limit Error
By setting SD Bus Power (SDBPWR) in SDMMC_PCR, the SDMMC is requested to supply power for the SD Bus. The SDMMC is protected from an illegal card by stopping power supply to the card, in which case this bit indicates a failure status. Reading 1 means the SDMMC is not supplying power to the card due to some failure. Reading 0 means that the SDMMC is supplying power and no error has occurred. The SDMMC may require some sampling time to detect the current limit.

This bit can only be set to '1' if SDMMC_EISTER.CURLIM is set to '1'. An interrupt can only be generated if SDMMC_EISIER.CURLIM is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 6 – DATEND** Data End Bit Error
This bit is set to '1' either when detecting 0 at the end bit position of read data which uses the DAT line or at the end bit position of the CRC Status.

This bit can only be set to '1' if SDMMC_EISTER.DATEND is set to '1'. An interrupt can only be generated if SDMMC_EISIER.DATEND is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 5 – DATCRC** Data CRC error
This bit is set to '1' when detecting a CRC error when transferring read data which uses the DAT line or when detecting that the Write CRC Status has a value other than '010'.

This bit can only be set to '1' if SDMMC_EISTER.DATCRC is set to '1'. An interrupt can only be generated if SDMMC_EISIER.DATCRC is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 4 – DATTEO** Data Timeout Error
This bit is set to '1' when detecting one of following timeout conditions.

– Busy timeout for R1b, R5b response type (see “Physical Layer Simplified Specification V3.01” and “SD Host Controller Simplified Specification V3.00”).

– Busy timeout after Write CRC status.

– Write CRC Status timeout.

– Read data timeout

This bit can only be set to '1' if SDMMC_EISTER.DATTEO is set to '1'. An interrupt can only be generated if SDMMC_EISIER.DATTEO is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 3 – CMDIDX** Command Index Error
This bit is set to '1' if a Command Index error occurs in the command response.

This bit can only be set to '1' if SDMMC_EISTER.CMDIDX is set to '1'. An interrupt can only be generated if SDMMC_EISIER.CMDIDX is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 2 – CMDEND** Command End Bit Error
This bit is set to '1' when detecting that the end bit of a command response is 0.

This bit can only be set to '1' if SDMMC_EISTER.CMDEND is set to '1'. An interrupt can only be generated if SDMMC_EISIER.CMDEND is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 1 – CMDCRC** Command CRC Error
The Command CRC Error is generated in two cases.

If a response is returned and the Command Timeout Error (CMDTEO) is set to 0 (indicating no command timeout), this bit is set to '1' when detecting a CRC error in the command response.

The SDMMC detects a CMD line conflict by monitoring the CMD line when a command is issued. If the SDMMC drives the CMD line to 1 level, but detects 0 level on the CMD line at the next SDCLK edge, then the SDMMC aborts the command (stops driving the CMD line) and sets this bit to '1'. CMDTEO is also set to '1' to indicate a CMD line conflict (see the table above).

This bit can only be set to '1' if SDMMC_EISTER.CMDCRC is set to '1'. An interrupt can only be generated if SDMMC_EISIER.CMDCRC is set to '1'.

Writing this bit to '1' clears this bit.

**Bit 0 – CMDTEO** Command Timeout Error
This bit is set to '1' only if no response is returned within 64 SDCLK cycles from the end bit of the command. If the SDMMC detects a CMD line conflict, in which case Command CRC Error (CMDCRC) is also set to '1' as shown in the table below, this bit is set without waiting for 64 SDCLK cycles because the command is aborted by the SDMMC.

This bit can only be set to '1' if SDMMC_EISTER.CMDTEO is set to '1'. An interrupt can only be generated if SDMMC_EISIER.CMDTEO is set to '1'.

Writing this bit to '1' clears this bit.

| CMDCRC | CMDTEO | Types of error |
|---|---|---|
| 0 | 0 | No error |
| 0 | 1 | Response timeout error |
| 1 | 0 | Response CRC error |
| 1 | 1 | CMD line conflict |

### 50.12.22 SDMMC Error Interrupt Status Register (e.MMC)

**Name:** SDMMC_EISTR (e.MMC)
**Offset:** 0x32
**Reset:** 0x0000
**Property:** Read/Write

**Note:** This register configuration is specific to the e.MMC operation mode.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | BOOTAE | | | ADMA | ACMD |
| Access | | | | R/W | | | R/W | R/W |
| Reset | | | | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CURLIM | DATEND | DATCRC | DATTEO | CMDIDX | CMDEND | CMDCRC | CMDTEO |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 12 – BOOTAE** Boot Acknowledge Error
This bit is set to '1' when detecting that the e.MMC Boot Acknowledge Status has a value other than '010'.

This bit can only be set to '1' if SDMMC_EISTER.BOOTAE is set to '1'. An interrupt can only be generated if SDMMC_EISIER.BOOTAE is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 9 – ADMA** ADMA Error
This bit is set to 1 when the SDMMC detects errors during an ADMA-based data transfer. The state of the ADMA at an error occurrence is saved in SDMMC_AESR.

In addition, the SDMMC raises this status flag when it detects some invalid description data (Valid = 0) at the ST_FDS state (see section “Advanced DMA” in the “SD Host Controller Simplified Specification V3.00” ). ADMA Error Status (ERRST) in SDMMC_AESR indicates that an error occurred in ST_FDS state. The user may find that the Valid bit is not set at the error descriptor.

This bit can only be set to '1' if SDMMC_EISTER.ADMA is set to '1'. An interrupt can only be generated if SDMMC_EISIER.ADMA is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 8 – ACMD** Auto CMD Error
Auto CMD12 and Auto CMD23 use this error status. This bit is set to '1' when detecting that one of the 0 to 4 bits in SDMMC_ACESR[4:0] has changed from 0 to 1. In the case of Auto CMD12, this bit is set to

'1', not only when errors occur in Auto CMD12, but also when Auto CMD12 is not executed due to the previous command error.

This bit can only be set to '1' if SDMMC_EISTER.ACMD is set to '1'. An interrupt can only be generated if SDMMC_EISIER.ACMD is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 7 – CURLIM** Current Limit Error
By setting SD Bus Power (SDBPWR) in SDMMC_PSR, the SDMMC is requested to supply power for the SD Bus. The SDMMC is protected from an illegal card by stopping power supply to the card, in which case this bit indicates a failure status. Reading 1 means the SDMMC is not supplying power to the card due to some failure. Reading 0 means that the SDMMC is supplying power and no error has occurred. The SDMMC may require some sampling time to detect the current limit.

This bit can only be set to '1' if SDMMC_EISTER.CURLIM is set to '1'. An interrupt can only be generated if SDMMC_EISIER.CURLIM is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 6 – DATEND** Data End Bit Error
This bit is set to '1' either when detecting 0 at the end bit position of read data which uses the DAT line or at the end bit position of the CRC Status.

This bit can only be set to '1' if SDMMC_EISTER.DATEND is set to '1'. An interrupt can only be generated if SDMMC_EISIER.DATEND is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 5 – DATCRC** Data CRC Error
This bit is set to '1' when detecting a CRC error during a transfer of read data which uses the DAT line or when detecting that the Write CRC Status has a value other than '010'.

This bit can only be set to '1' if SDMMC_EISTER.DATCRC is set to '1'. An interrupt can only be generated if SDMMC_EISIER. DATCRC is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 4 – DATTEO** Data Timeout error
This bit is set to '1' when detecting one of following timeout conditions.

– Busy timeout for R1b, R5b response type (see “Physical Layer Simplified Specification V3.01” and “SDIO Simplified Specification V3.00” ).

– Busy timeout after Write CRC Status.

– Write CRC Status timeout.

– Read data timeout

This bit can only be set to '1' if SDMMC_EISTER.DATTEO is set to '1'. An interrupt can only be generated if SDMMC_EISIER.DATTEO is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 3 – CMDIDX** Command Index Error
This bit is set to '1' if a Command Index error occurs in the command response.

This bit can only be set to '1' if SDMMC_EISTER.CMDIDX is set to '1'. An interrupt can only be generated if SDMMC_EISIER.CMDIDX is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 2 – CMDEND** Command End Bit Error
This bit is set to '1' when detecting that the end bit of a command response is 0.

This bit can only be set to '1' if SDMMC_EISTER.CMDEND is set to '1'. An interrupt can only be generated if SDMMC_EISIER.CMDEND is set to '1'.

Writing this bit to '1' clears this bit.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 1 – CMDCRC** Command CRC Error
The Command CRC Error is generated in two cases.

If a response is returned and Command Timeout Error (CMDTEO) is set to 0 (indicating no command timeout), this bit is set to '1' when detecting a CRC error in the command response.

The SDMMC detects a CMD line conflict by monitoring the CMD line when a command is issued. If the SDMMC drives the CMD line to 1 level, but detects 0 level on the CMD line at the next SDCLK edge, then the SDMMC aborts the command (stops driving the CMD line) and sets this bit to '1'. CMDTEO is also set to '1' to indicate a CMD line conflict (see he table “Relations between CMDCRC and CMDTEO”).

This bit can only be set to '1' if SDMMC_EISTER.CMDCRC is set to '1'. An interrupt can only be generated if SDMMC_EISIER.CMDCRC is set to '1'.

Writing this bit to 1 clears this bit.

**Bit 0 – CMDTEO** Command Timeout Error
This bit is set to '1' only if no response is returned within 64 SDCLK cycles from the end bit of the command. If the SDMMC detects a CMD line conflict, in which case Command CRC Error (CMDCRC) is also set to '1' as shown in the table “Relations between CMDCRC and CMDTEO”, this bit is set without waiting for 64 SDCLK cycles because the command is aborted by the SDMMC.

This bit can only be set to '1' if SDMMC_EISTER.CMDTEO is set to '1'. An interrupt can only be generated if SDMMC_EISIER.CMDTEO is set to '1'.

Writing this bit to '1' clears this bit.

| CMDCRC | CMDTEO | Types of error |
|---|---|---|
| 0 | 0 | No error |
| 0 | 1 | Response timeout error |
| 1 | 0 | Response CRC error |
| 1 | 1 | CMD line conflict |

### 50.12.23 SDMMC Normal Interrupt Status Enable Register (SD_SDIO)

**Name:** SDMMC_NISTER (SD_SDIO)
**Offset:** 0x34
**Reset:** 0x0000
**Property:** Read/Write

**Note:** This register configuration is specific to the SD/SDIO operation mode.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CINT |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CREM | CINS | BRDRDY | BWRRDY | DMAINT | BLKGE | TRFC | CMDC |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 8 – CINT** Card Interrupt Status Enable
If this bit is set to 0, the SDMMC clears interrupt requests to the system. The Card Interrupt detection is stopped when this bit is cleared and restarted when this bit is set to 1. The user may clear this bit before servicing the Card Interrupt and may set this bit again after all interrupt requests from the card are cleared to prevent inadvertent interrupts.

0 (MASKED): The CINT status flag in SDMMC_NISTR is masked.

1 (ENABLED): The CINT status flag in SDMMC_NISTR is enabled.

**Bit 7 – CREM** Card Removal Status Enable
0 (MASKED): The CREM status flag in SDMMC_NISTR is masked.

1 (ENABLED): The CREM status flag in SDMMC_NISTR is enabled.

**Bit 6 – CINS** Card Insertion Status Enable
0 (MASKED): The CINS status flag in SDMMC_NISTR is masked.

1 (ENABLED): The CINS status flag in SDMMC_NISTR is enabled.

**Bit 5 – BRDRDY** Buffer Read Ready Status Enable
0 (MASKED): The BRDRDY status flag in SDMMC_NISTR is masked.

1 (ENABLED): The BRDRDY status flag in SDMMC_NISTR is enabled.

**Bit 4 – BWRRDY** Buffer Write Ready Status Enable
0 (MASKED): The BWRRDY status flag in SDMMC_NISTR is masked.

1 (ENABLED): The BWRRDY status flag in SDMMC_NISTR is enabled.

**Bit 3 – DMAINT** DMA Interrupt Status Enable
0 (MASKED): The DMAINT status flag in SDMMC_NISTR is masked.

1 (ENABLED): The DMAINT status flag in SDMMC_NISTR is enabled.

**Bit 2 – BLKGE** Block Gap Event Status Enable
0 (MASKED): The BLKGE status flag in SDMMC_NISTR is masked.

1 (ENABLED): The BLKGE status flag in SDMMC_NISTR is enabled.

**Bit 1 – TRFC** Transfer Complete Status Enable
0 (MASKED): The TRFC status flag in SDMMC_NISTR is masked.

1 (ENABLED): The TRFC status flag in SDMMC_NISTR is enabled.

**Bit 0 – CMDC** Command Complete Status Enable
0 (MASKED): The CMDC status flag in SDMMC_NISTR is masked.

1 (ENABLED): The CMDC status flag in SDMMC_NISTR is enabled.

### 50.12.24 SDMMC Normal Interrupt Status Enable Register (e.MMC)

**Name:** SDMMC_NISTER (e.MMC)
**Offset:** 0x34
**Reset:** 0x0000
**Property:** Read/Write

**Note:** This register configuration is specific to the e.MMC operation mode.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | BOOTAR | | | | | | |
| Access | | R/W | | | | | | |
| Reset | | 0 | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | BRDRDY | BWRRDY | DMAINT | BLKGE | TRFC | CMDC |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 14 – BOOTAR** Boot Acknowledge Received Status Enable
0 (MASKED): The BOOTAR status flag in SDMMC_NISTR is masked.

1 (ENABLED): The BOOTAR status flag in SDMMC_NISTR is enabled.

**Bit 5 – BRDRDY** Buffer Read Ready Status Enable
0 (MASKED): The BRDRDY status flag in SDMMC_NISTR is masked.

1 (ENABLED): The BRDRDY status flag in SDMMC_NISTR is enabled.

**Bit 4 – BWRRDY** Buffer Write Ready Status Enable
0 (MASKED): The BWRRDY status flag in SDMMC_NISTR is masked.

1 (ENABLED): The BWRRDY status flag in SDMMC_NISTR is enabled.

**Bit 3 – DMAINT** DMA Interrupt Status Enable
0 (MASKED): The DMAINT status flag in SDMMC_NISTR is masked.

1 (ENABLED): The DMAINT status flag in SDMMC_NISTR is enabled.

**Bit 2 – BLKGE** Block Gap Event Status Enable
0 (MASKED): The BLKGE status flag in SDMMC_NISTR is masked.

1 (ENABLED): The BLKGE status flag in SDMMC_NISTR is enabled.

**Bit 1 – TRFC** Transfer Complete Status Enable
0 (MASKED): The TRFC status flag in SDMMC_NISTR is masked.

1 (ENABLED): The TRFC status flag in SDMMC_NISTR is enabled.

**Bit 0 – CMDC** Command Complete Status Enable
0 (MASKED): The CMDC status flag in SDMMC_NISTR is masked.

1 (ENABLED): The CMDC status flag in SDMMC_NISTR is enabled.

### 50.12.25 SDMMC Error Interrupt Status Enable Register (SD_SDIO)

**Name:** SDMMC_EISTER (SD_SDIO)
**Offset:** 0x36
**Reset:** 0x0000
**Property:** Read/Write

**Note:** This register configuration is specific to the SD/SDIO operation mode.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ADMA | ACMD |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CURLIM | DATEND | DATCRC | DATTEO | CMDIDX | CMDEND | CMDCRC | CMDTEO |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 9 – ADMA** ADMA Error Status Enable
0 (MASKED): The ADMA status flag in SDMMC_EISTR is masked.

1 (ENABLED): The ADMA status flag in SDMMC_EISTR is enabled.

**Bit 8 – ACMD** Auto CMD Error Status Enable
0 (MASKED): The ACMD status flag in SDMMC_EISTR is masked.

1 (ENABLED): The ACMD status flag in SDMMC_EISTR is enabled.

**Bit 7 – CURLIM** Current Limit Error Status Enable
0 (MASKED): The CURLIM status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CURLIM status flag in SDMMC_EISTR is enabled.

**Bit 6 – DATEND** Data End Bit Error Status Enable
0 (MASKED): The DATEND status flag in SDMMC_EISTR is masked.

1 (ENABLED): The DATEND status flag in SDMMC_EISTR is enabled.

**Bit 5 – DATCRC** Data CRC Error Status Enable
0 (MASKED): The DATCRC status flag in SDMMC_EISTR is masked.

1 (ENABLED): The DATCRC status flag in SDMMC_EISTR is enabled.

**Bit 4 – DATTEO** Data Timeout Error Status Enable
0 (MASKED): The DATTEO status flag in SDMMC_EISTR is masked.

1 (ENABLED): The DATTEO status flag in SDMMC_EISTR is enabled.

**Bit 3 – CMDIDX** Command Index Error Status Enable
0 (MASKED): The CMDIDX status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDIDX status flag in SDMMC_EISTR is enabled.

**Bit 2 – CMDEND** Command End Bit Error Status Enable
0 (MASKED): The CMDEND status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDEND status flag in SDMMC_EISTR is enabled.

**Bit 1 – CMDCRC** Command CRC Error Status Enable
0 (MASKED): The CMDCRC status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDCRC status flag in SDMMC_EISTR is enabled.

**Bit 0 – CMDTEO** Command Timeout Error Status Enable
0 (MASKED): The CMDTEO status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDTEO status flag in SDMMC_EISTR is enabled.

### 50.12.26 SDMMC Error Interrupt Status Enable Register (e.MMC)

**Name:** SDMMC_EISTER (e.MMC)
**Offset:** 0x36
**Reset:** 0x0000
**Property:** Read/Write

**Note:** This register configuration is specific to the e.MMC operation mode.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | BOOTAE | | | ADMA | ACMD |
| Access | | | | R/W | | | R/W | R/W |
| Reset | | | | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CURLIM | DATEND | DATCRC | DATTEO | CMDIDX | CMDEND | CMDCRC | CMDTEO |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 12 – BOOTAE** Boot Acknowledge Error Status Enable
0 (MASKED): The BOOTAE status flag in SDMMC_EISTR is masked.

1 (ENABLED): The BOOTAE status flag in SDMMC_EISTR is enabled.

**Bit 9 – ADMA** ADMA Error Status Enable
0 (MASKED): The ADMA status flag in SDMMC_EISTR is masked.

1 (ENABLED): The ADMA status flag in SDMMC_EISTR is enabled.

**Bit 8 – ACMD** Auto CMD Error Status Enable
0 (MASKED): The ACMD status flag in SDMMC_EISTR is masked.

1 (ENABLED): The ACMD status flag in SDMMC_EISTR is enabled.

**Bit 7 – CURLIM** Current Limit Error Status Enable
0 (MASKED): The CURLIM status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CURLIM status flag in SDMMC_EISTR is enabled.

**Bit 6 – DATEND** Data End Bit Error Status Enable
0 (MASKED): The DATEND status flag in SDMMC_EISTR is masked.

1 (ENABLED): The DATEND status flag in SDMMC_EISTR is enabled.

**Bit 5 – DATCRC** Data CRC Error Status Enable
0 (MASKED): The DATCRC status flag in SDMMC_EISTR is masked.

1 (ENABLED): The DATCRC status flag in SDMMC_EISTR is enabled.

**Bit 4 – DATTEO** Data Timeout Error Status Enable
0 (MASKED): The DATTEO status flag in SDMMC_EISTR is masked.

1 (ENABLED): The DATTEO status flag in SDMMC_EISTR is enabled.

**Bit 3 – CMDIDX** Command Index Error Status Enable
0 (MASKED): The CMDIDX status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDIDX status flag in SDMMC_EISTR is enabled.

**Bit 2 – CMDEND** Command End Bit Error Status Enable
0 (MASKED): The CMDEND status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDEND status flag in SDMMC_EISTR is enabled.

**Bit 1 – CMDCRC** Command CRC Error Status Enable
0 (MASKED): The CMDCRC status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDCRC status flag in SDMMC_EISTR is enabled.

**Bit 0 – CMDTEO** Command Timeout Error Status Enable
0 (MASKED): The CMDTEO status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDTEO status flag in SDMMC_EISTR is enabled.

### 50.12.27 SDMMC Normal Interrupt Signal Enable Register (SD_SDIO)

**Name:** SDMMC_NISIER (SD_SDIO)
**Offset:** 0x38
**Reset:** 0x0000
**Property:** Read/Write

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CINT |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CREM | CINS | BRDRDY | BWRRDY | DMAINT | BLKGE | TRFC | CMDC |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 8 – CINT** Card Interrupt Signal Enable
0 (MASKED): No interrupt is generated when the CINT status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the CINT status rises in SDMMC_NISTR.

**Bit 7 – CREM** Card Removal Signal Enable
0 (MASKED): No interrupt is generated when the CREM status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the CREM status rises in SDMMC_NISTR.

**Bit 6 – CINS** Card Insertion Signal Enable
0 (MASKED): No interrupt is generated when the CINS status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the CINS status rises in SDMMC_NISTR.

**Bit 5 – BRDRDY** Buffer Read Ready Signal Enable
0 (MASKED): No interrupt is generated when the BRDRDY status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the BRDRDY status rises in SDMMC_NISTR.

**Bit 4 – BWRRDY** Buffer Write Ready Signal Enable
0 (MASKED): No interrupt is generated when the BWRRDY status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the BWRRDY status rises in SDMMC_NISTR.

**Bit 3 – DMAINT** DMA Interrupt Signal Enable
0 (MASKED): No interrupt is generated when the DMAINT status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the DMAINT status rises in SDMMC_NISTR.

**Bit 2 – BLKGE** Block Gap Event Signal Enable
0 (MASKED): No interrupt is generated when the BLKGE status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the BLKGE status rises in SDMMC_NISTR.

**Bit 1 – TRFC** Transfer Complete Signal Enable
0 (MASKED): No interrupt is generated when the TRFC status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the TRFC status rises in SDMMC_NISTR.

**Bit 0 – CMDC** Command Complete Signal Enable
0 (MASKED): No interrupt is generated when the CMDC status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the CMDC status rises in SDMMC_NISTR.

### 50.12.28 SDMMC Normal Interrupt Signal Enable Register (e.MMC)

**Name:** SDMMC_NISIER (e.MMC)
**Offset:** 0x38
**Reset:** 0x0000
**Property:** Read/Write

**Note:** This register configuration is specific to the e.MMC operation mode.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | BOOTAR | | | | | | |
| Access | | R/W | | | | | | |
| Reset | | 0 | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | BRDRDY | BWRRDY | DMAINT | BLKGE | TRFC | CMDC |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 14 – BOOTAR** Boot Acknowledge Received Signal Enable
0 (MASKED): No interrupt is generated when the BOOTAR status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the BOOTAR status rises in SDMMC_NISTR.

**Bit 5 – BRDRDY** Buffer Read Ready Signal Enable
0 (MASKED): No interrupt is generated when the BRDRDY status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the BRDRDY status rises in SDMMC_NISTR.

**Bit 4 – BWRRDY** Buffer Write Ready Signal Enable
0 (MASKED): No interrupt is generated when the BWRRDY status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the BWRRDY status rises in SDMMC_NISTR.

**Bit 3 – DMAINT** DMA Interrupt Signal Enable
0 (MASKED): No interrupt is generated when the DMAINT status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the DMAINT status rises in SDMMC_NISTR.

**Bit 2 – BLKGE** Block Gap Event Signal Enable
0 (MASKED): No interrupt is generated when the BLKGE status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the BLKGE status rises in SDMMC_NISTR.

**Bit 1 – TRFC** Transfer Complete Signal Enable
0 (MASKED): No interrupt is generated when the TRFC status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the TRFC status rises in SDMMC_NISTR.

**Bit 0 – CMDC** Command Complete Signal Enable
0 (MASKED): No interrupt is generated when the CMDC status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the CMDC status rises in SDMMC_NISTR.

### 50.12.29 SDMMC Error Interrupt Signal Enable Register (SD_SDIO)

**Name:** SDMMC_EISIER (SD_SDIO)
**Offset:** 0x3A
**Reset:** 0x0000
**Property:** Read/Write

**Note:** This register configuration is specific to the SD/SDIO operation mode.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ADMA | ACMD |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CURLIM | DATEND | DATCRC | DATTEO | CMDIDX | CMDEND | CMDCRC | CMDTEO |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 9 – ADMA** ADMA Error Signal Enable
0 (MASKED): No interrupt is generated when the ADMA status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the ADMA status rises in SDMMC_EISTR.

**Bit 8 – ACMD** Auto CMD Error Signal Enable
0 (MASKED): No interrupt is generated when the ACMD status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the ACMD status rises in SDMMC_EISTR.

**Bit 7 – CURLIM** Current Limit Error Signal Enable
0 (MASKED): No interrupt is generated when the CURLIM status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CURLIM status rises in SDMMC_EISTR.

**Bit 6 – DATEND** Data End Bit Error Signal Enable
0 (MASKED): No interrupt is generated when the DATEND status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the DATEND status rises in SDMMC_EISTR.

**Bit 5 – DATCRC** Data CRC Error Signal Enable
0 (MASKED): No interrupt is generated when the DATCRC status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the DATCRC status rises in SDMMC_EISTR.

**Bit 4 – DATTEO** Data Timeout Error Signal Enable
0 (MASKED): No interrupt is generated when the DATTEO status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the DATTEO status rises in SDMMC_EISTR.

**Bit 3 – CMDIDX** Command Index Error Signal Enable
0 (MASKED): No interrupt is generated when the CMDIDX status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CMDIDX status rises in SDMMC_EISTR.

**Bit 2 – CMDEND** Command End Bit Error Signal Enable
0 (MASKED): No interrupt is generated when the CMDEND status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CMDEND status rises in SDMMC_EISTR.

**Bit 1 – CMDCRC** Command CRC Error Signal Enable
0 (MASKED): No interrupt is generated when the CDMCRC status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CMDCRC status rises in SDMMC_EISTR.

**Bit 0 – CMDTEO** Command Timeout Error Signal Enable
0 (MASKED): No interrupt is generated when the CMDTEO status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CMDTEO status rises in SDMMC_EISTR.

### 50.12.30 SDMMC Error Interrupt Signal Enable Register (e.MMC)

**Name:** SDMMC_EISIER (e.MMC)
**Offset:** 0x3A
**Reset:** 0x0000
**Property:** Read/Write

**Note:** This register configuration is specific to the e.MMC operation mode.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | BOOTAE | | | ADMA | ACMD |
| Access | | | | | | | | |
| Reset | | | | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CURLIM | DATEND | DATCRC | DATTEO | CMDIDX | CMDEND | CMDCRC | CMDTEO |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 12 – BOOTAE** Boot Acknowledge Error Signal Enable
0 (MASKED): No interrupt is generated when the BOOTAE status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the BOOTAE status rises in SDMMC_EISTR.

**Bit 9 – ADMA** ADMA Error Signal Enable
0 (MASKED): No interrupt is generated when the ADMA status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the ADMA status rises in SDMMC_EISTR.

**Bit 8 – ACMD** Auto CMD Error Signal Enable
0 (MASKED): No interrupt is generated when the ACMD status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the ACMD status rises in SDMMC_EISTR.

**Bit 7 – CURLIM** Current Limit Error Signal Enable
0 (MASKED): No interrupt is generated when the CURLIM status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CURLIM status rises in SDMMC_EISTR.

**Bit 6 – DATEND** Data End Bit Error Signal Enable
0 (MASKED): No interrupt is generated when the DATEND status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the DATEND status rises in SDMMC_EISTR.

**Bit 5 – DATCRC** Data CRC Error Signal Enable
0 (MASKED): No interrupt is generated when the DATCRC status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the DATCRC status rises in SDMMC_EISTR.

**Bit 4 – DATTEO** Data Timeout Error Signal Enable
0 (MASKED): No interrupt is generated when the DATTEO status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the DATTEO status rises in SDMMC_EISTR.

**Bit 3 – CMDIDX** Command Index Error Signal Enable
0 (MASKED): No interrupt is generated when the CMDIDX status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CMDIDX status rises in SDMMC_EISTR.

**Bit 2 – CMDEND** Command End Bit Error Signal Enable
0 (MASKED): No interrupt is generated when the CMDEND status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CMDEND status rises in SDMMC_EISTR.

**Bit 1 – CMDCRC** Command CRC Error Signal Enable
0 (MASKED): No interrupt is generated when the CDMCRC status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CMDCRC status rises in SDMMC_EISTR.

**Bit 0 – CMDTEO** Command Timeout Error Signal Enable
0 (MASKED): No interrupt is generated when the CMDTEO status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CMDTEO status rises in SDMMC_EISTR.

### 50.12.31 SDMMC Auto CMD Error Status Register

**Name:** SDMMC_ACESR
**Offset:** 0x3C
**Reset:** 0x0000
**Property:** Read-only

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMDNI | | | ACMDIDX | ACMDEND | ACMDCRC | ACMDTEO | ACMD12NE |
| Access | R | | | R | R | R | R | R |
| Reset | 0 | | | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – CMDNI** Command Not Issued by Auto CMD12 Error
This bit is set to 1 when CMD_wo_DAT is not executed due to an Auto CMD12 error (SDMMC_ACESR[4:1]). This bit is set to 0 when Auto CMD Error is generated by Auto CMD23.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 4 – ACMDIDX** Auto CMD Index Error
This bit is set to 1 when the Command Index error occurs in response to a command.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 3 – ACMDEND** Auto CMD End Bit Error
This bit is set to 1 when detecting that the end bit of the command response is 0.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bit 2 – ACMDCRC** Auto CMD CRC Error
This bit is set to 1 when detecting a CRC error in the command response (see the table above for more details).

**Bit 1 – ACMDTEO** Auto CMD Timeout Error
This bit is set to 1 if no response is returned within 64 SDCLK cycles from the end bit of the command. If this bit is set to 1, the other error status bits (SDMMC_ACESR[4:2]) are meaningless.

**Table 50-4. Relation between ACMDCRC and ACMDTEO**

| ACMDCRC | ACMDTEO | Types of error |
|---|---|---|
| 0 | 0 | No error |
| 0 | 1 | Response Timeout error |
| 1 | 0 | Response CRC error |
| 1 | 1 | CMD line conflict |

**Bit 0 – ACMD12NE** Auto CMD12 Not Executed
If a memory multiple block data transfer is not started due to a command error, this bit is not set to 1 because it is not necessary to issue Auto CMD12. Setting this bit to 1 means the SDMMC cannot issue Auto CMD12 to stop a memory multiple block data transfer due to some error. If this bit is set to 1, other error status bits (SDMMC_ACESR[4:1]) are meaningless.

This bit is set to 0 when an Auto CMD error is generated by Auto CMD23.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

### 50.12.32 SDMMC Host Control 2 Register (SD_SDIO)

**Name:** SDMMC_HC2R (SD_SDIO)
**Offset:** 0x3E
**Reset:** 0x0000
**Property:** Read/Write

**Note:** This register configuration is specific to the SD/SDIO operation mode.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PVALEN | ASINTEN | | | | | | |
| Access | R/W | R/W | | | | | | |
| Reset | 0 | 0 | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SCLKSEL | EXTUN | DRVSEL[1:0] | | VS18EN | UHSMS[2:0] | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 15 – PVALEN** Preset Value Enable
As the operating SDCLK frequency and I/O driver strength depend on the system implementation, it is difficult to determine these parameters in the standard host driver. When PVALEN is set to 1, automatic SDCLK frequency generation and driver strength selection are performed without considering system-specific conditions. This bit enables the functions defined in SDMMC_PVR.

If this bit is set to 0, SDCLKFSEL, CLKGSEL in SDMMC_CCR and DRVSEL in SDMMC_HC2R are set by the user.

If this bit is set to 1, SDCLKFSEL, CLKGSEL in SDMMC_CCR and DRVSEL in SDMMC_HC2R are set by the SDMMC as specified in SDMMC_PVR.

| Value | Description |
|---|---|
| 0 | SDCLK and Driver strength are controlled by the user. |
| 1 | Automatic selection by Preset Value is enabled. |

**Bit 14 – ASINTEN** Asynchronous Interrupt Enable
This bit can be set to 1 if a card supports asynchronous interrupts and Asynchronous Interrupt Support (ASINTSUP) is set to 1 in SDMMC_CA0R. Asynchronous interrupt is effective when DAT[1] interrupt is used in 4-bit SD mode. If this bit is set to 1, the user can stop the SDCLK during the asynchronous interrupt period to save power. During this period, the SDMMC continues to deliver the Card Interrupt to the host when it is asserted by the card.

| Value | Description |
|---|---|
| 0 | Disabled |
| 1 | Enabled |

**Bit 7 – SCLKSEL** Sampling Clock Select
The SDMMC uses this bit to select the sampling clock to receive CMD and DAT.

This bit is set by the tuning procedure and is valid after completion of tuning (when EXTUN is cleared). Setting 1 means that tuning is completed successfully and setting 0 means that tuning has failed.

Writing 1 to this bit is meaningless and ignored. A tuning circuit is reset by writing to 0. This bit can be cleared by setting EXTUN to 1. Once the tuning circuit is reset, it takes time to complete the tuning sequence. Therefore, the user should keep this bit to 1 to perform a retuning sequence to complete a retuning sequence in a short time. Changing this bit is not allowed while the SDMMC is receiving a response or a read data block. See Figure 2.29 in the “SD Host Controller Simplified Specification V3.00”.

| Value | Description |
|---|---|
| 0 | The fixed clock is used to sample data. |
| 1 | The tuned clock is used to sample data. |

**Bit 6 – EXTUN** Execute Tuning
This bit is set to 1 to start the tuning procedure and is automatically cleared when the tuning procedure is completed. The result of tuning is indicated to Sampling Clock Select (SCLKSEL). The tuning procedure is aborted by writing 0. See Figure 2.29 in the “SD Host Controller Simplified Specification V3.00”.

| Value | Description |
|---|---|
| 0 | Not tuned or tuning completed. |
| 1 | Execute tuning. |

**Bits 5:4 – DRVSEL[1:0]** Driver Strength Select
The SDMMC output driver in 1.8V signaling is selected by this bit. In 3.3V signaling, this field is not effective. This field can be set according to the Driver Type A, C and D support bits in SDMMC_CA1R.

This field depends on the setting of Preset Value Enable (PVALEN):

– PVALEN = 0: This field is set by the user.

– PVALEN = 1: This field is automatically set by a value specified in one of the SDMMC_PVRx.

| Value | Name | Description |
|---|---|---|
| 0 | TYPEB | Driver Type B is selected (Default) |
| 1 | TYPEA | Driver Type A is selected |
| 2 | TYPEC | Driver Type C is selected |
| 3 | TYPED | Driver Type D is selected |

**Bit 3 – VS18EN** 1.8V Signaling Enable
This bit controls the SDMMC_1V8SEL output, which in turn may control an external voltage regulator for the I/O cell and card I/Os. 3.3V or some other fixed voltage is supplied to the card/device regardless of the signaling voltage.

Setting this bit from 0 to 1 starts changing the signal voltage from 3.3V to 1.8V. The 1.8V regulator output must be stable within 5 ms.

Clearing this bit from 1 to 0 starts changing the signal voltage from 1.8V to 3.3V. The 3.3V regulator output must be stable within 5 ms.

The user can set this bit to 1 when the SDMMC supports 1.8V signaling (one of the support bits is set to 1: SDR50SUP, SDR104SUP or DDR50SUP in SDMMC_CA1R) and the card or device supports UHS-I (S18A = 1. See “Bus Switch Voltage Switch Sequence in the “Physical Layer Simplified Specification V3.01” ).

| Value | Description |
|---|---|
| 0 | 3.3V signaling. |
| 1 | 1.8V signaling. |

**Bits 2:0 – UHSMS[2:0]** UHS Mode Select
This field is used to select one of the UHS-I modes and is effective when 1.8V Signal Enable (VS18EN) is set to 1.

This field is effective only if SDMMC_MC1R.DDR is set to 0.

If Preset Value Enable is set to 1, the SDMMC sets SDCLK Frequency Select (SDCLKFSEL), Clock Generator Select (CLKGSEL) in SDMMC_CCR and Driver Strength Select (DRVSEL) according to SDMMC_PVR. In this case, one of the preset value registers is selected by this field. The user needs to reset SD Clock Enable (SDCLKEN) before changing this field to avoid generating a clock glitch. After setting this field, the user sets SDCLKEN to 1 again.

| Value | Name | Description |
|---|---|---|
| 0 | SDR12 | UHS SDR12 Mode |
| 1 | SDR25 | UHS SDR25 Mode |
| 2 | SDR50 | UHS SDR50 Mode |
| 3 | SDR104 | UHS SDR104 Mode |
| 4 | DDR50 | UHS DDR50 Mode |

### 50.12.33 SDMMC Host Control 2 Register (e.MMC)

**Name:** SDMMC_HC2R (e.MMC)
**Offset:** 0x3E
**Reset:** 0x0000
**Property:** Read/Write

**Note:** This register configuration is specific to the e.MMC operation mode.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PVALEN | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SCLKSEL | EXTUN | DRVSEL[1:0] | | HS200EN[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 15 – PVALEN** Preset Value Enable
As the operating SDCLK frequency and I/O driver strength depend on the system implementation, it is difficult to determine these parameters in the standard host driver. When Preset Value Enable (PVALEN) is set to 1, automatic SDCLK frequency generation and driver strength selection are performed without considering system-specific conditions. This bit enables the functions defined in SDMMC_PVR.

If this bit is set to 0, SDCLKFSEL, CLKGSEL in SDMMC_CCR and DRVSEL in SDMMC_HC2R are set by the user.

If this bit is set to 1, SDCLKFSEL, CLKGSEL in SDMMC_CCR and DRVSEL in SDMMC_HC2R are set by the SDMMC as specified in SDMMC_PVR.

| Value | Description |
|---|---|
| 0 | SDCLK and Driver strength are controlled by the user. |
| 1 | Automatic selection by Preset Value is enabled. |

**Bit 7 – SCLKSEL** Sampling Clock Select
The SDMMC uses this bit to select the sampling clock to receive CMD and DAT.

This bit is set by the tuning procedure and is valid after completion of tuning (when EXTUN is cleared). Setting 1 means that tuning is completed successfully and setting 0 means that tuning has failed.

Writing 1 to this bit is meaningless and ignored. A tuning circuit is reset by writing to 0. This bit can be cleared by setting EXTUN to 1. Once the tuning circuit is reset, it takes time to complete a tuning sequence. Therefore, the user should keep this bit to 1 to perform a retuning sequence to complete a retuning sequence in a short time. Changing this bit is not allowed while the SDMMC is receiving a response or a read data block. See Figure 2.29 in the “SD Host Controller Simplified Specification V3.00”.

| Value | Description |
|---|---|
| 0 | The fixed clock is used to sample data. |
| 1 | The tuned clock is used to sample data. |

**Bit 6 – EXTUN** Execute Tuning
This bit is set to 1 to start the tuning procedure and is automatically cleared when the tuning procedure is completed. The result of tuning is indicated to Sampling Clock Select (SCLKSEL). The tuning procedure is aborted by writing 0. See Figure 2.29 in the “SD Host Controller Simplified Specification V3.00”.

| Value | Description |
|---|---|
| 0 | Not tuned or tuning completed |
| 1 | Execute tuning |

**Bits 5:4 – DRVSEL[1:0]** Driver Strength Select
The SDMMC output driver in 1.8V signaling is selected by this bit. In 3.3V signaling, this field is not effective. This field can be set according to the Driver Type A, C and D support bits in SDMMC_CA1R.

This field depends on setting of Preset Value Enable (PVALEN):

– PVALEN = 0: This field is set by the user.

– PVALEN = 1: This field is automatically set by a value specified in one of the SDMMC_PVRx.

| Value | Name | Description |
|---|---|---|
| 0 | TYPEB | Driver Type B is selected (Default) |
| 1 | TYPEA | Driver Type A is selected |
| 2 | TYPEC | Driver Type C is selected |
| 3 | TYPED | Driver Type D is selected |

**Bits 3:0 – HS200EN[3:0]** HS200 Mode Enable
This field is used to select the e.MMC HS200 mode. When HS200EN is set to $B_{(hexa)}$, the HS200 mode is enabled. Any other value except 0 is forbidden when interfacing an e.MMC device.

If Preset Value Enable is set to 1, SDMMC sets SDCLK Frequency Select (SDCLKFSEL), Clock Generator Select (CLKGSEL) in SDMMC_CCR and Driver Strength Select (DRVSEL) according to SDMMC_PVR. In this case, one of the preset value registers is selected by this field. The user needs to reset SD Clock Enable (SDCLKEN) before changing this field to avoid generating a clock glitch. After setting this field, the user sets SDCLKEN to 1 again.

Note: This field is effective only if DDR in SDMMC_MC1R is set to 0.

### 50.12.34 SDMMC Capabilities 0 Register

**Name:** SDMMC_CA0R
**Offset:** 0x40
**Property:** Read/Write

**Reset:** Register reset values depend on the instance of the SDMMC:

| Instance | Reset Value |
|---|---|
| SDMMC0 | 0x27EC0C8C |
| SDMMC1 | 0x27E80C8C |

**Note:** The Capabilities 0 Register is not supposed to be written by the user. However, the user can modify preset values only if Capabilities Write Enable (CAPWREN) is set to 1 in SDMMC_CACR.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SLTYPE[1:0] | | ASINTSUP | SB64SUP | | V18VSUP | V30VSUP | V33VSUP |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SRSUP | SDMASUP | HSSUP | | ADMA2SUP | ED8SUP | MAXBLKL[1:0] | |
| Access | R/W | R/W | R/W | | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BASECLKF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TEOCLKU | | TEOCLKF[5:0] | | | | | |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

**Bits 31:30 – SLTYPE[1:0]** Slot Type
This field indicates usage of a slot by a specific system. An SDMMC control register set is defined per slot.

Embedded Slot for One Device means that only one nonremovable device is connected to a bus slot.

The Standard Host Driver controls a removable card (SLTYPE = 0) or one embedded device (SLTYPE = 1) connected to an SD bus slot.

| Value | Name |
|---|---|
| 0 | Removable Card Slot |
| 1 | Embedded Slot for One Device |
| 2 | Reserved |
| 3 | Reserved |

**Bit 29 – ASINTSUP** Asynchronous Interrupt Support
See section “Asynchronous Interrupt” in the “SDIO Simplified Specification V3.00” .

| Value | Description |
|---|---|
| 0 | Asynchronous interrupt not supported. |
| 1 | Asynchronous interrupt supported. |

**Bit 28 – SB64SUP** 64-Bit System Bus Support
Reading this bit to 1 means that the SDMMC supports the 64-bit Address Descriptor mode and is connected to the 64-bit address system bus.

| Value | Description |
|---|---|
| 0 | 64-bit address bus not supported. |
| 1 | 64-bit address bus supported. |

**Bit 26 – V18VSUP** Voltage Support 1.8V

| Value | Description |
|---|---|
| 0 | 1.8V Voltage supply not supported. |
| 1 | 1.8V Voltage supply supported. |

**Bit 25 – V30VSUP** Voltage Support 3.0V

| Value | Description |
|---|---|
| 0 | 3.0V Voltage supply not supported. |
| 1 | 3.0V Voltage supply supported. |

**Bit 24 – V33VSUP** Voltage Support 3.3V

| Value | Description |
|---|---|
| 0 | 3.3V Voltage supply not supported. |
| 1 | 3.3V Voltage supply supported. |

**Bit 23 – SRSUP** Suspend/Resume Support
This bit indicates whether the SDMMC supports the Suspend/Resume functionality. If this bit is set to 0, the user does not issue either Suspend or Resume commands because the Suspend and Resume mechanism (see “Suspend and Resume Mechanism” in the “SD Host Controller Simplified Specification V3.00” ) is not supported.

| Value | Description |
|---|---|
| 0 | Suspend/Resume not supported. |
| 1 | Suspend/Resume supported. |

**Bit 22 – SDMASUP** SDMA Support
This bit indicates whether the SDMMC is capable of using SDMA to transfer data between system memory and the SDMMC directly.

| Value | Description |
|---|---|
| 0 | SDMA not supported. |
| 1 | SDMA supported. |

**Bit 21 – HSSUP** High Speed Support
This bit indicates whether the SDMMC and the system support High Speed mode and they can supply SDCLK frequency from 25 MHz to 50 MHz.

| Value | Description |
|---|---|
| 0 | High Speed not supported. |
| 1 | High Speed supported. |

**Bit 19 – ADMA2SUP** ADMA2 Support
This bit indicates whether the SDMMC is capable of using ADMA2.

| Value | Description |
|---|---|
| 0 | ADMA2 not supported. |
| 1 | ADMA2 supported. |

**Bit 18 – ED8SUP** 8-Bit Support for Embedded Device
This bit indicates whether the SDMMC is capable of using the 8-bit Bus Width mode.

| Value | Description |
|---|---|
| 0 | 8-bit bus width not supported. |
| 1 | 8-bit bus width supported. |

**Bits 17:16 – MAXBLKL[1:0]** Max Block Length
This field indicates the maximum block size that the user can read and write to the buffer in the SDMMC. Three sizes can be defined, as shown below. It is noted that the transfer block length is always 512 bytes for SD Memory Cards regardless of this field.

| Value | Name | Description |
|---|---|---|
| 0 | 512 | 512 bytes |
| 1 | 1024 | 1024 bytes |
| 2 | 2048 | 2048 bytes |
| 3 | NONE | Reserved |

**Bits 15:8 – BASECLKF[7:0]** Base Clock Frequency
This value indicates the frequency of the base clock (BASECLK). The user uses this value to calculate the clock divider value (see SDCLK Frequency Select (SDCLKFSEL) in SDMMC_CCR).

If this field is set to 0, the user must get the information via another method.

$f_{\mathrm{BASECLK}} = \mathrm{BASECLKF}_{\mathrm{MHz}}$

**Bit 7 – TEOCLKU** Timeout Clock Unit
This bit shows the unit of the base clock frequency used to detect Data Timeout Error.

| Value | Description |
|---|---|
| 0 | KHz |
| 1 | MHz |

**Bits 5:0 – TEOCLKF[5:0]** Timeout Clock Frequency
This bit shows the timeout clock frequency (TEOCLK) used to detect Data Timeout Error.

If this field is set to 0, the user must get the information via another method.

The Timeout Clock Unit (TEOCLKU) defines the unit of this field's value.

– TEOCLKU = 0: $f_{\mathrm{TEOCLK}} = \mathrm{TEOCLKF}_{\mathrm{KHz}}$

– TEOCLKU = 1: $f_{\mathrm{TEOCLK}} = \mathrm{TEOCLKF}_{\mathrm{MHz}}$

### 50.12.35 SDMMC Capabilities 1 Register

**Name:** SDMMC_CA1R
**Offset:** 0x44
**Property:** Read/Write

**Note:** The Capabilities 1 Register is not supposed to be written by the user. However, the user can modify preset values only if Capabilities Write Enable (CAPWREN) is set to 1 in SDMMC_CACR.

**Note:** Register reset values depend on the instance of the SDMMC:

| Instance | Reset Value |
|---|---|
| SDMMC0 | 0x00200F77 |
| SDMMC1 | 0x00200070 |

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CLKMULT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RTMOD[1:0] | | TSDR50 | | TCNTRT[3:0] | | | |
| Access | R/W | R/W | R/W | | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | DRVDSUP | DRVCSUP | DRVASUP | | DDR50SUP | SDR104SUP | SDR50SUP |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | | | | | | |

**Bits 23:16 – CLKMULT[7:0]** Clock Multiplier
This field indicates the multiplier factor between the Base Clock (BASECLK) used for the Divided Clock Mode and the Multiplied Clock (MULTCLK) used for the Programmable Clock mode (see SDMMC_CCR).

Reading this field to 0 means that the Programmable Clock mode is not supported.

$f_{\mathrm{MULTCLK}} = f_{\mathrm{BASECLK}} \times (\mathrm{CLKMULT} + 1)$

**Bits 15:14 – RTMOD[1:0]** Retuning Modes
This field selects the retuning method and limits the maximum data length.

There are two retuning timings: Retuning Request (RTREQ) controlled by the SDMMC, and expiration of a Retuning timer controlled by the user. By receiving either timing, the user executes the retuning procedure just before a next command issue.

The maximum data length per read/write command is restricted so that retuning procedures can be inserted during data transfers.

**Retuning Mode 1:**

The SDMMC does not have any internal logic to detect when retuning needs to be performed. In this case, the user should maintain all retuning timings by using the Retuning Timer. To enable inserting the retuning procedure during data transfers, the data length per Read/Write command must be limited to 4 Mbytes.

**Retuning Mode 2:**

The SDMMC has the capability to indicate the retuning timing by Retuning Request (RTREQ) during data transfers. Then the data length per Read/Write command must be limited to 4 Mbytes.

During nondata transfer, retuning timing is determined by either Retuning Request or Retuning Timer. If Retuning Request is used, Retuning Timer should be disabled.

**Retuning Mode 3:**

The SDMMC has the capability to take care of the retuning during data transfer (Auto Retuning). Retuning Request is not generated during data transfers and there is no limitation to data length per Read/Write command.

During nondata transfer, retuning timing is determined either by Retuning Request or Retuning Timer. If Retuning Request is used, Retuning Timer should be disabled.

| Value | Name | Description | Data Length |
|---|---|---|---|
| 0 | MODE1 | Timer | 4 Mbytes (Max) |
| 1 | MODE2 | Timer and Retuning Request | 4 Mbytes (Max) |
| 2 | MODE3 | Auto Retuning (for transfer) Timer and Retuning Request | Any |
| 3 | – | Reserved | – |

**Bit 13 – TSDR50** Use Tuning for SDR50
If this bit is set to 1, the SDMMC requires tuning to operate SDR50 (tuning is always required to operate SDR104).

| Value | Description |
|---|---|
| 0 | SDR50 does not require tuning. |
| 1 | SDR50 requires tuning. |

**Bits 11:8 – TCNTRT[3:0]** Timer Count For Retuning
This field indicates an initial value of the Retuning Timer for Retuning Mode (RTMOD) 1 to 3. Reading this field at 0 means that the Retuning Timer is disabled. The Retuning Timer initial value ranges from 0 to 1024 seconds.

$t_{TIMER} = 2^{(TCNTRT - 1)}$Seconds

**Bit 6 – DRVDSUP** Driver Type D Support

| Value | Description |
|---|---|
| 0 | Driver type D is not supported. |
| 1 | Driver type D is supported. |

**Bit 5 – DRVCSUP** Driver Type C Support

| Value | Description |
|---|---|
| 0 | Driver type C is not supported. |
| 1 | Driver type C is supported. |

**Bit 4 – DRVASUP** Driver Type A Support

| Value | Description |
|---|---|
| 0 | Driver type A is not supported. |
| 1 | Driver type A is supported. |

**Bit 2 – DDR50SUP** DDR50 Support

| Value | Description |
|---|---|
| 0 | DDR50 mode is not supported. |
| 1 | DDR50 mode is supported. |

**Bit 1 – SDR104SUP** SDR104 Support

| Value | Description |
|---|---|
| 0 | SDR104 mode is not supported. |
| 1 | SDR104 mode is supported. |

**Bit 0 – SDR50SUP** SDR50 Support

| Value | Description |
|---|---|
| 0 | SDR50 mode is not supported. |
| 1 | SDR50 mode is supported. |

### 50.12.36 SDMMC Maximum Current Capabilities Register

**Name:** SDMMC_MCCAR
**Offset:** 0x48
**Reset:** 0x00000000
**Property:** Read/Write

**Note:** The Maximum Current Capabilities Register is not supposed to be written by the user. However, the user can modify preset values only if Capabilities Write Enable (CAPWREN) is set to 1 in SDMMC_CACR.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MAXCUR18V[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MAXCUR30V[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MAXCUR33V[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – MAXCUR18V[7:0]** Maximum Current for 1.8V
This field indicates the maximum current capability for 1.8V voltage. This value is meaningful only if V18VSUP is set to 1 in SDMMC_CA0R. Reading MAXCUR18V at 0 means that the user must get information via another method.

$\mathrm{Imax}_{\mathrm{mA}} = 4 \times \mathrm{MAXCURR18}V$

**Bits 15:8 – MAXCUR30V[7:0]** Maximum Current for 3.0V
This field indicates the maximum current capability for 3.0V voltage. This value is meaningful only if V30VSUP is set to 1 in SDMMC_CA0R. Reading MAXCUR30V at 0 means that the user must get information via another method.

$\mathrm{Imax}_{\mathrm{mA}} = 4 \times \mathrm{MAXCURR30}V$

**Bits 7:0 – MAXCUR33V[7:0]** Maximum Current for 3.3V
This field indicates the maximum current capability for 3.3V voltage. This value is meaningful only if V33VSUP is set to 1 in SDMMC_CA0R. Reading MAXCUR33V at 0 means that the user must get information via another method.

$\mathrm{Imax}_{\mathrm{mA}} = 4 \times \mathrm{MAXCURR33}V$

### 50.12.37 SDMMC Force Event Register for Auto CMD Error Status

**Name:** SDMMC_FERACES
**Offset:** 0x50
**Reset:** –
**Property:** Write-only

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMDNI | | | ACMDIDX | ACMDEND | ACMDCRC | ACMDTEO | ACMD12NE |
| Access | W | | | W | W | W | W | W |
| Reset | – | | | – | – | – | – | – |

**Bit 7 – CMDNI** Force Event for Command Not Issued by Auto CMD12 Error
For test purposes, the user can write this bit to 1 to raise the CMDNI status flag in SDMMC_ACESR.

Writing this bit to 0 has no effect.

**Bit 4 – ACMDIDX** Force Event for Auto CMD Index Error
For test purposes, the user can write this bit to 1 to raise the ACMDIDX status flag in SDMMC_ACESR.

Writing this bit to 0 has no effect.

**Bit 3 – ACMDEND** Force Event for Auto CMD End Bit Error
For test purposes, the user can write this bit to 1 to raise the ACMDEND status flag in SDMMC_ACESR.

Writing this bit to 0 has no effect.

**Bit 2 – ACMDCRC** Force Event for Auto CMD CRC Error
For test purposes, the user can write this bit to 1 to raise the ACMDCRC status flag in SDMMC_ACESR.

Writing this bit to 0 has no effect.

**Bit 1 – ACMDTEO** Force Event for Auto CMD Timeout Error
For test purposes, the user can write this bit to 1 to raise the ACMDTEO status flag in SDMMC_ACESR.

Writing this bit to 0 has no effect.

**Bit 0 – ACMD12NE** Force Event for Auto CMD12 Not Executed
For test purposes, the user can write this bit to 1 to raise the ACMD12NE status flag in SDMMC_ACESR.

Writing this bit to 0 has no effect.

### 50.12.38 SDMMC Force Event Register for Error Interrupt Status

**Name:** SDMMC_FEREIS
**Offset:** 0x52
**Reset:** –
**Property:** Write-only

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | BOOTAE | | | ADMA | ACMD |
| Access | | | | W | | | W | W |
| Reset | | | | – | | | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CURLIM | DATEND | DATCRC | DATTEO | CMDIDX | CMDEND | CMDCRC | CMDTEO |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 12 – BOOTAE** Force Event for Boot Acknowledge Error
For test purposes, the user can write this bit to 1 to raise the BOOTAE status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

**Bit 9 – ADMA** Force Event for ADMA Error
For test purposes, the user can write this bit to 1 to raise the ADMA status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

**Bit 8 – ACMD** Force Event for Auto CMD Error
For test purposes, the user can write this bit to 1 to raise the ACMD status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

**Bit 7 – CURLIM** Force Event for Current Limit Error
For test purposes, the user can write this bit to 1 to raise the CURLIM status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

**Bit 6 – DATEND** Force Event for Data End Bit Error
For test purposes, the user can write this bit to 1 to raise the DATEND status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

**Bit 5 – DATCRC** Force Event for Data CRC error
For test purposes, the user can write this bit to 1 to raise the DATCRC status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

**Bit 4 – DATTEO** Force Event for Data Timeout error
For test purposes, the user can write this bit to 1 to raise the DATTEO status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

**Bit 3 – CMDIDX** Force Event for Command Index Error
For test purposes, the user can write this bit to 1 to raise the CMDIDX status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

**Bit 2 – CMDEND** Force Event for Command End Bit Error
For test purposes, the user can write this bit to 1 to raise the CDMEND status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

**Bit 1 – CMDCRC** Force Event for Command CRC Error
For test purposes, the user can write this bit to 1 to raise the CMDCRC status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

**Bit 0 – CMDTEO** Force Event for Command Timeout Error
For test purposes, the user can write this bit to 1 to raise the CMDTEO status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

### 50.12.39 SDMMC ADMA Error Status Register

**Name:** SDMMC_AESR
**Offset:** 0x54
**Reset:** 0x00
**Property:** Read-only

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | LMIS | ERRST[1:0] | |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – LMIS** ADMA Length Mismatch Error
This error occurs in the following two cases:

– While Block Count Enable (BCEN) is being set, the total data length specified by the Descriptor table is different from that specified by the Block Count (BLKCNT) and Transfer Block Size (BLKSIZE).

– The total data length cannot be divided by the Transfer Block Size (BLKSIZE).

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | Error. |

**Bits 1:0 – ERRST[1:0]** ADMA Error State
This field indicates the state of ADMA when an error has occurred during an ADMA data transfer. This field never reads 2 because ADMA never stops in this state.

| Value | Name | Description |
|---|---|---|
| 0 | ST_STOP (Stop DMA) | Points to the descriptor following the error descriptor |
| 1 | ST_FDS (Fetch Descriptor) | Points to the error descriptor |
| 2 | – | (Not used) |
| 3 | ST_TRF (Transfer Data) | Points to the descriptor following the error descriptor |

### 50.12.40 SDMMC ADMA System Address Register 0

**Name:** SDMMC_ASAR0
**Offset:** 0x58
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADMASA[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADMASA[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADMASA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADMASA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADMASA[31:0]** ADMA System Address
This field holds the byte address of the executing command of the descriptor table. The 32-bit address descriptor uses SDMMC_ASAR. At the start of ADMA, the user must set the start address of the descriptor table. The ADMA increments this register address, which points to the next Descriptor line to be fetched.

When the ADMA Error (ADMA) status flag rises, this field holds a valid descriptor address depending on the ADMA Error State (ERRST). The user must program Descriptor Table on 32-bit boundary and set 32-bit boundary address to this register. ADMA2 ignores the lower 2 bits of this register and assumes it to be 0.

### 50.12.41 SDMMC Preset Value Register

**Name:** SDMMC_PVRx
**Offset:** 0x60 + x*0x02 [x=0..7]
**Reset:** 0x00000000
**Property:** Read/Write

One of the Preset Value Registers is effective based on the selected bus speed mode. The table below defines the conditions to select one of the SDMMC_PVRs.

**Table 50-5. Preset Value Register Select Condition**

| Selected Bus Speed Mode | VS18EN (SDMMC_HC2R) | HSEN (SDMMC_HC1R) | UHSMS (SDMMC_HC2R) |
|---|---|---|---|
| Default Speed | 0 | 0 | don't care |
| High Speed | 0 | 1 | don't care |
| SDR12 | 1 | don't care | 0 |
| SDR25 | 1 | don't care | 1 |
| SDR50 | 1 | don't care | 2 |
| SDR104/HS200 | 1 | don't care | 3 |
| DDR50 | 1 | don't care | 4 |
| Reserved | 1 | don't care | Other values |

The table below shows the effective Preset Value Register according to the Selected Bus Speed mode.

**Table 50-6. Preset Value Registers**

| SDMMC_PVRx | Selected Bus Speed Mode | Signal Voltage |
|---|---|---|
| SDMMC_PVR0 | Initialization | 3.3V or 1.8V |
| SDMMC_PVR1 | Default Speed | 3.3V |
| SDMMC_PVR2 | High Speed | 3.3V |
| SDMMC_PVR3 | SDR12 | 1.8V |
| SDMMC_PVR4 | SDR25 | 1.8V |
| SDMMC_PVR5 | SDR50 | 1.8V |
| SDMMC_PVR6 | SDR104/HS200 | 1.8V |
| SDMMC_PVR7 | DDR50 | 1.8V |

When Preset Value Enable (PVALEN) in SDMMC_HC2R is set to 1, SDCLK Frequency Select (SDLCKFSEL) and Clock Generator Select (CLKGSEL) in SDMMC_CCR, and Driver Strength Select (DRVSEL) in SDMMC_HC2R are automatically set based on the Selected Bus Speed mode. This means that the user does not need to set these fields when preset is enabled. A Preset Value Register for Initialization (SDMMC_PVR0) is not selected by Bus Speed mode. Before starting the initialization

sequence, the user needs to set a clock preset value to SDCLKFSEL in SDMMC_CCR. PVALEN can be set to 1 after the initialization is completed.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DRVSEL[1:0] | | | | | CLKGSEL | SDCLKFSEL[9:8] | |
| Access | R/W | R/W | | | | R/W | R/W | R/W |
| Reset | 0 | 0 | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SDCLKFSEL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:14 – DRVSEL[1:0]** Driver Strength Select
See DRVSEL in SDMMC_HC2R.

**Bit 10 – CLKGSEL** Clock Generator Select
See CLKGSEL in SDMMC_CCR.

**Bits 9:0 – SDCLKFSEL[9:0]** SDCLK Frequency Select
See SDCLKFSEL in SDMMC_CCR.

### 50.12.42 SDMMC Slot Interrupt Status Register

**Name:** SDMMC_SISR
**Offset:** 0xFC
**Reset:** 0x0000
**Property:** Read-only

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | INTSSL[1:0] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

**Bits 1:0 – INTSSL[1:0]** Interrupt Signal for Each Slot
These status bits indicate the logical OR of Interrupt Signals and Wakeup Signal for each SDMMC instance in the product (INTSSL[x] corresponds to SDMMCx instance in the product).

### 50.12.43 SDMMC Host Controller Version Register

**Name:** SDMMC_HCVR
**Offset:** 0xFE
**Reset:** 0x1502
**Property:** Read-only

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | VVER[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SVER[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

**Bits 15:8 – VVER[7:0]** Vendor Version Number
Reserved. Value subject to change. No functionality associated. This is the internal version of the module.

**Bits 7:0 – SVER[7:0]** Specification Version Number
This status indicates the SD Host Controller Specification Version.

| Value | Name |
|---|---|
| 0 | SD Host Specification Version 1.00 |
| 1 | SD Host Specification Version 2.00, including the feature of the ADMA and Test Register |
| 2 | SD Host Specification Version 3.00 |

### 50.12.44 SDMMC Additional Present State Register

**Name:** SDMMC_APSR
**Offset:** 0x200
**Reset:** –
**Property:** Read-only

**Reset:**

Reset value depends on the instance of the SDMMC:

| Instance | Reset Value |
|---|---|
| SDMMC0 | 0x0000000F |
| SDMMC1 | 0x00000000 |

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | HDATLL[3:0] | | | |
| Access | | | | | R | R | R | R |
| Reset | | | | | – | – | – | – |

**Bits 3:0 – HDATLL[3:0]** DAT[7:4] High Line Level
This status is used to check the DAT[7:4] line level to recover from errors, and for debugging.

### 50.12.45 SDMMC e.MMC Control 1 Register

**Name:** SDMMC_MC1R
**Offset:** 0x204
**Reset:** 0x00
**Property:** Read/Write

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FCD | RSTN | BOOTA | OPD | DDR | | CMDTYP[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | | 0 | 0 |

**Bit 7 – FCD** e.MMC Force Card Detect
When using e.MMC, the user can set this bit to 1 to bypass the card detection procedure using the SDMMC_CD signal.

0 (DISABLED): e.MMC Forced Card Detect is disabled. The SDMMC_CD signal is used and debounce timing is applied.

1 (ENABLED): e.MMC Forced Card Detect is enabled.

**Bit 6 – RSTN** e.MMC Reset Signal
This bit controls the e.MMC reset signal.

| Value | Description |
|---|---|
| 0 | Reset signal is inactive. |
| 1 | Reset signal is active. |

**Bit 5 – BOOTA** e.MMC Boot Acknowledge Enable
This bit must be set according to the value of BOOT_ACK in the Extended CSD Register (see “Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51” ).

When this bit is set to 1, the SDMMC waits for boot acknowledge pattern from the e.MMC before receiving boot data.

If the boot acknowledge pattern is wrong, the BOOTAE status flag rises in SDMMC_EISTR if BOOTAE is set in SDMMC_EISTER. An interrupt is generated if BOOTAE is set in SDMMC_EISIER.

If the no boot acknowledge pattern is received, the DATTEO status flag rises in SDMMC_EISTR if DATTEO is set in SDMMC_EISTER. An interrupt is generated if DATTEO is set in SDMMC_EISIER.

**Bit 4 – OPD** e.MMC Open Drain Mode
This bit sets the command line in open drain.

| Value | Description |
|---|---|
| 0 | The command line is in push-pull. |
| 1 | The command line is in open drain. |

**Bit 3 – DDR** e.MMC HSDDR Mode
This bit selects the High Speed DDR mode.

The clock divider (DIV) in SDMMC_CCR must be set to a value different from 0 when DDR is 1.

| Value | Description |
|---|---|
| 0 | High Speed DDR is not selected. |
| 1 | High Speed DDR is selected. |

**Bits 1:0 – CMDTYP[1:0]** e.MMC Command Type

| Value | Name | Description |
|---|---|---|
| 0 | NORMAL | The command is not an e.MMC specific command. |
| 1 | WAITIRQ | This bit must be set to 1 when the e.MMC is in Interrupt mode (CMD40). See “Interrupt Mode” in the “Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51” . |
| 2 | STREAM | This bit must be set to 1 in the case of Stream Read(CMD11) or Stream Write (CMD20). Only effective for e.MMC up to revision 4.41. |
| 3 | BOOT | Starts a Boot Operation mode at the next write to SDMMC_CR. Boot data are read directly from e.MMC device. |

### 50.12.46 SDMMC e.MMC Control 2 Register

**Name:** SDMMC_MC2R
**Offset:** 0x205
**Reset:** –
**Property:** Write-only

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ABOOT | SRESP |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

**Bit 1 – ABOOT** e.MMC Abort Boot
This bit is used to exit from Boot mode. Writing this bit to 1 exits the Boot Operation mode. Writing 0 is ignored.

**Bit 0 – SRESP** e.MMC Abort Wait IRQ
This bit is used to exit from the Interrupt mode. When this bit is written to 1, the SDMMC sends the CMD40 response automatically. This brings the e.MMC from Interrupt mode to the standard Data Transfer mode. Writing this bit to 0 is ignored.

This bit is only effective when CMD_TYP in SDMMC_MC1R is set to WAITIRQ.

### 50.12.47 SDMMC AHB Control Register

**Name:** SDMMC_ACR
**Offset:** 0x208
**Reset:** 0x00
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | BMAX[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bits 1:0 – BMAX[1:0]** AHB Maximum Burst
This field selects the maximum burst size in case of DMA transfer.

| Value | Name | Description |
|---|---|---|
| 0 | INCR16 | The maximum burst size is INCR16. |
| 1 | INCR8 | The maximum burst size is INCR8. |
| 2 | INCR4 | The maximum burst size is INCR4. |
| 3 | SINGLE | Only SINGLE transfers are performed. |

### 50.12.48 SDMMC Clock Control 2 Register

**Name:** SDMMC_CC2R
**Offset:** 0x20C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | FSDCLKD |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – FSDCLKD** Force SDCLK Disabled
The user can choose to maintain the SDCLK during 8 SDCLK cycles after the end bit of the last data block in case of a read transaction, or after the end bit of the CRC status in case of a write transaction.

| Value | Description |
|---|---|
| 0 | The SDCLK is forced and it cannot be stopped immediately after the transaction. |
| 1 | The SDCLK is not forced and it can be stopped immediately after the transaction. |

### 50.12.49 SDMMC Retuning Control 1 Register

**Name:** SDMMC_RTC1R
**Offset:** 0x210
**Reset:** 0x00
**Property:** Read/Write

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | TMREN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – TMREN** Retuning Timer Enable
Enable the retuning timer.

0 (DISABLED): The retuning timer is disabled.

1 (ENABLED): The retuning timer is enabled.

### 50.12.50 SDMMC Retuning Control 2 Register

**Name:** SDMMC_RTC2R
**Offset:** 0x211
**Reset:** –
**Property:** Write-only

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | RLD |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – RLD** Retuning Timer Reload
This bit is only efficient if the Retuning timer is enabled (SDMMC_RTC1R.TMREN set to 1). Once the Timer Counter Value (TCVAL) is set to a nonzero value in SDMMC_RTCVR, setting this bit to 1 starts the timer count. The retuning timer count restarts each time this bit is written to 1.

Writing this bit to 0 has no effect.

### 50.12.51 SDMMC Retuning Counter Value Register

**Name:** SDMMC_RTCVR
**Offset:** 0x214
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TCVAL[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 3:0 – TCVAL[3:0]** Retuning Timer Counter Value
The TCVAL value is used to define the time before expiration of the retuning timer where:

$$\text{Time} = 2^{\text{TCVAL} - 1}\text{seconds}$$

This value must range between 1 and 11. Any other value results in the retuning timer being disabled.

### 50.12.52 SDMMC Retuning Interrupt Status Enable Register

**Name:** SDMMC_RTISTER
**Offset:** 0x218
**Reset:** 0x00
**Property:** Read/Write

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | TEVT |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – TEVT** Retuning Timer Event
0 (MASKED): The TEVT status flag in SDMMC_RTISTR is masked.

1 (ENABLED): The TEVT status flag in SDMMC_RTISTR is enabled.

### 50.12.53 SDMMC Retuning Interrupt Signal Enable Register

**Name:** SDMMC_RTISIER
**Offset:** 0x219
**Reset:** 0x00
**Property:** Read/Write

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | TEVT |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – TEVT** Retuning Timer Event
0 (MASKED): No interrupt is generated when the TEVT status rises in SDMMC_RTISTR.

1 (ENABLED): An interrupt is generated when the TEVT status rises in SDMMC_RTISTR.

### 50.12.54 SDMMC Retuning Interrupt Status Register

**Name:** SDMMC_RTISTR
**Offset:** 0x21C
**Reset:** 0x00
**Property:** Read/Write

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | TEVT |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – TEVT** Retuning Timer Event
This bit is set to 1 when the retuning timer count is elapsed if SDMMC_RTISTER.TEVT is set to 1. An interrupt is generated if SDMMC_RTISIER.TEVT is set to 1.

Writing this bit to 1 clears this bit.

| Value | Description |
|---|---|
| 0 | No retuning timer event. |
| 1 | Retuning timer event. |

### 50.12.55 SDMMC Retuning Status Slots Register

**Name:** SDMMC_RTSSR
**Offset:** 0x21D
**Reset:** 0x00
**Property:** Read-only

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | TEVTSLOT[1:0] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

**Bits 1:0 – TEVTSLOT[1:0]** Retuning Timer Event Slots
Indicates the TEVT status for each SDMMC instance in the product (TEVTSLOT[x] corresponds to SDMMCx instance in the product).

### 50.12.56 SDMMC Tuning Control Register

**Name:** SDMMC_TUNCR
**Offset:** 0x220
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SMPLPT |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – SMPLPT** Sampling Point
This bit selects the position of the sampling point into the data window for SDR104 and HS200 modes.

| Value | Description |
|---|---|
| 0 | Sampling point is set at 50% of the data window. |
| 1 | Sampling point is set at 75% of the data window. |

### 50.12.57 SDMMC Capabilities Control Register

**Name:** SDMMC_CACR
**Offset:** 0x230
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | KEY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CAPWREN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 15:8 – KEY[7:0]** Key

| Value | Name | Description |
|---|---|---|
| 46h | KEY | Writing any other value in this field aborts the write operation of the CAPWREN bit.<br>Always reads as 0. |

**Bit 0 – CAPWREN** Capabilities Write Enable
This bit can only be written if the value of KEY is 46h.

| Value | Description |
|---|---|
| 0 | Capabilities registers (SDMMC_CA0R, SDMMC_CA1R and SDMMC_CA1R) cannot be written. |
| 1 | Capabilities registers (SDMMC_CA0R, SDMMC_CA1R and SDMMC_CA1R) can be written. |

### 50.12.58 SDMMC Calibration Control Register

**Name:** SDMMC_CALCR
**Offset:** 0x240
**Reset:** 0x0000500E
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CALPBP[3:0] | | | | CALP[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CALNBP[3:0] | | | | CALN[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CNTVAL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | BPEN | TUNDIS | ALWYSON | CLKDIV[2:0] | | | EN |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 1 | 1 | 1 | 0 |

**Bits 31:28 – CALPBP[3:0]** Calibration P Bypass Value
Calibration code applied for the p-channel transistors when BPEN is set to 1. This field is ignored if BPEN is 0.

**Bits 27:24 – CALP[3:0]** Calibration P Status
Calibration code for the p-channel transistors to match the required output impedance.

**Bits 23:20 – CALNBP[3:0]** Calibration N Bypass Value
Calibration code applied for the n-channel transistors when BPEN is set to 1. This field is ignored if BPEN is 0.

**Bits 19:16 – CALN[3:0]** Calibration N Status
Calibration code for the n-channel transistors to match the required output impedance.

**Bits 15:8 – CNTVAL[7:0]** Calibration Counter Value
Defines the number of XXXX cycles (divided by 4) required to cover the I/O calibration cell startup time.

$$\mathrm{CNTVAL}_{\mathrm{Minimum}} = \frac{t_{\mathrm{STARTUP}}}{4 \times t_{\mathrm{HCLOCK}}}$$

$t_{STARTUP}$ = 2 µs

**Bit 6 – BPEN** Calibration Bypass Enabled

| Value | Description |
|---|---|
| 0 | Calibration bypass is not enabled. |
| 1 | Calibration bypass is enabled. CALPBP and CALNBP codes are applied to the calibration cell. |

**Bit 5 – TUNDIS** Calibration During Tuning Disabled

| Value | Description |
|---|---|
| 0 | Calibration is launched before each tuning. |
| 1 | Calibration is not launched at tuning. |

**Bit 4 – ALWYSON** Calibration Analog Always ON

| Value | Description |
|---|---|
| 0 | Calibration analog is shut down after each calibration. |
| 1 | Calibration analog remains powered after calibration. |

**Bits 3:1 – CLKDIV[2:0]** Calibration Clock Division
The clock applied to the calibration cell is divided by CLKDIV + 1

**Bit 0 – EN** PADs Calibration Enable

| Value | Description |
|---|---|
| 0 | SDMMC I/O calibration disabled. |
| 1 | SDMMC I/O calibration enabled. |

# 51. Image Sensor Controller (ISC)

## 51.1 Description

The Image Sensor Controller (ISC) system manages incoming data from a parallel sensor. It supports a single active interface. The parallel interface protocol can use a free-running clock or a gated clock strategy. It supports the ITU-R BT 656/1120 422 protocol with a data width of 8 bits or 10 bits and raw Bayer format. The internal image processor includes adjustable white balance, color filter array interpolation, color correction, gamma correction, 12 bits to 10 bits compression, programmable color space conversion, horizontal and vertical chrominance subsampling module. The module also integrates a triple channel direct memory access controller master interface.

## 51.2 Embedded Characteristics

- Parallel 12-bit Interface for Raw Bayer, YCbCr, Monochrome and JPEG Compressed Sensor Interface
- BT.601/656/1120 Video Interface Supported
- Progressive Systems and Segmented Frame Systems
- Raw Bayer, YCbCr, Luminance (Black and White) Pixel Format Supported
- Resolution up to 2592 x 1944
- Input Pixel Clock up to 96 MHz
- Output Master Clock Generation
- Cropping
- Adjustable White Balance
- Raw Bayer Color Filter Array Interpolation
- Color Correction
- Gamma Correction
- Color Space Conversion
- Contrast and Brightness Control
- 4:4:4 to 4:2:2 Subsampler
- 4:2:2 to 4:2:0 Subsampler
- Rounding, Limiting and Packing unit
- Histogram Generation
- System Interface: Direct Memory Access Interface with Packed, Semi Planar and Planar output format
- Output Memory Format: 16 bpp RGB, 32 bpp RGB, 16 bpp, YCbCr 444, YCbCr 422, YCbCr 420, up to 12 bit raw Bayer

© 2018 Microchip Technology Inc.

## 51.3 Block Diagram and Use Cases

### 51.3.1 Image Sensor Controller Functional Diagrams

**Figure 51-1. ISC Block Diagram**

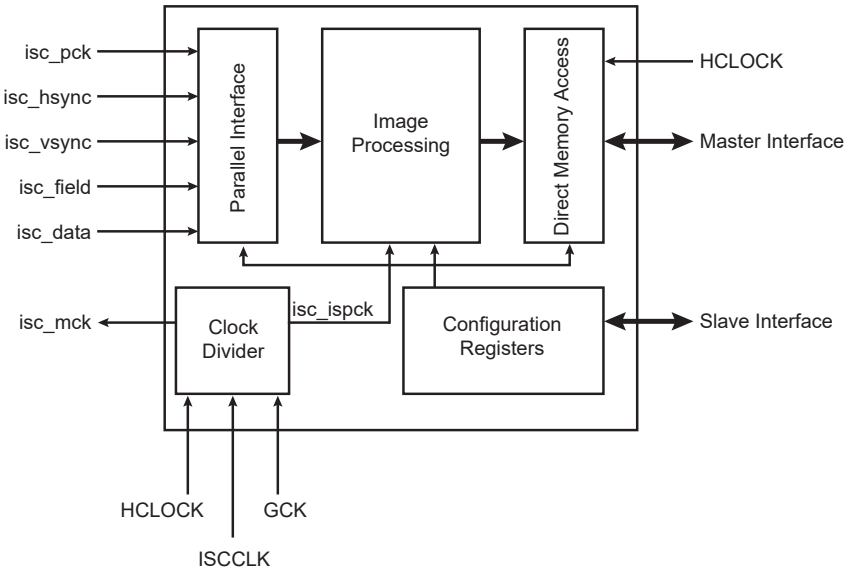
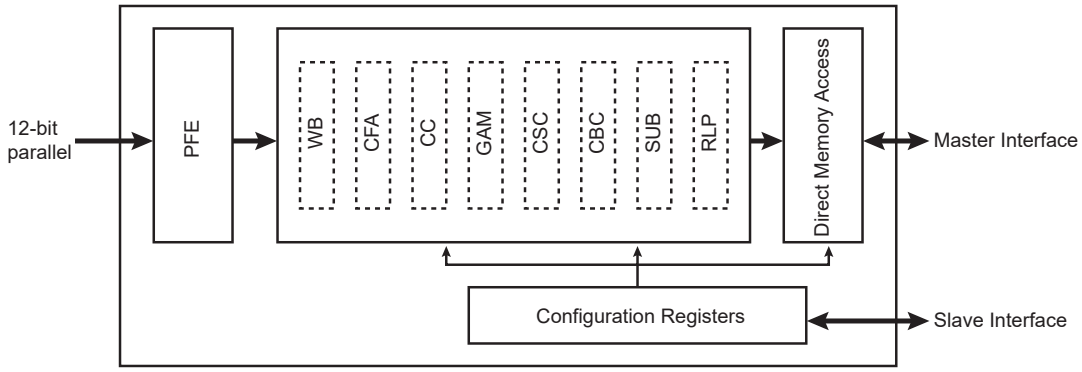



**Figure 51-2. ISC Raw Bayer Signal Processor**

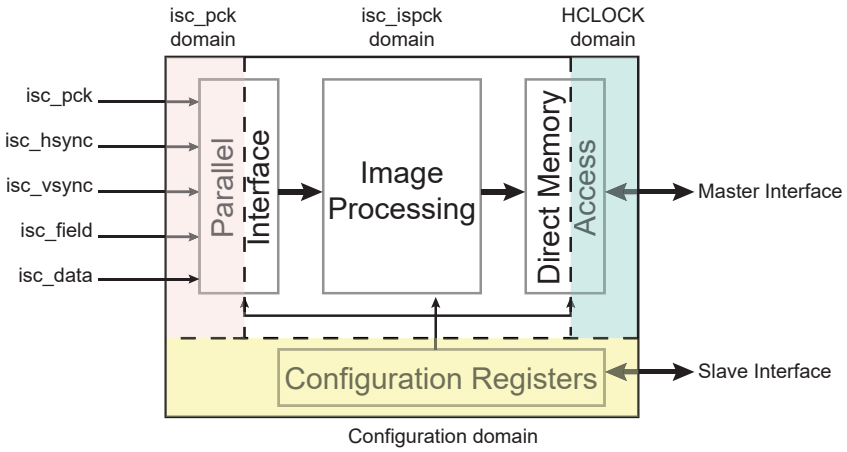



The ISC video pipeline integrates the following submodules:

- PFE: Parallel Front End to sample the camera sensor input stream
- WB: Programmable white balance in the Bayer domain
- CFA: Color filter array interpolation module
- CC: Programmable color correction
- GAM: Gamma correction
- CSC: Programmable color space conversion
- CBC: Contrast and Brightness control
- SUB: This module performs YCbCr444 to YCbCr420 chrominance subsampling.
- RLP: This module performs rounding, range limiting and packing of the incoming data.

### 51.3.2 Image Sensor Controller Clock Domain Diagram

**Figure 51-3. Clock Domain Hierarchy**




### 51.3.3 Image Sensor Controller Typical Use Cases

**Figure 51-4. Raw Bayer Sensor**

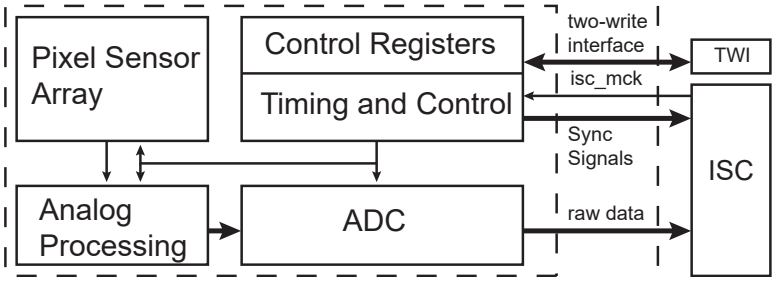



**Figure 51-5. Raw Bayer Sensor with Embedded Image Processor**

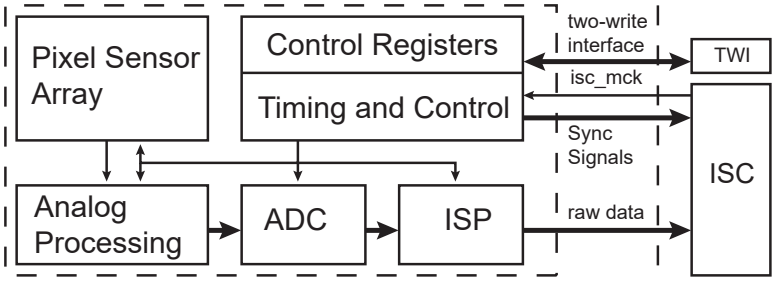



**Figure 51-6. BT656 Video Interface Sensor**

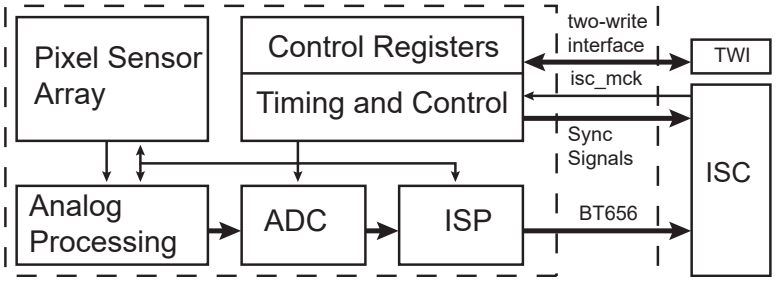


**Figure 51-7. Sensor with JPEG Output**

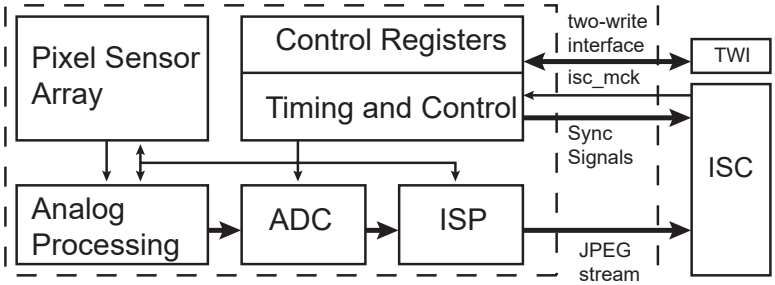
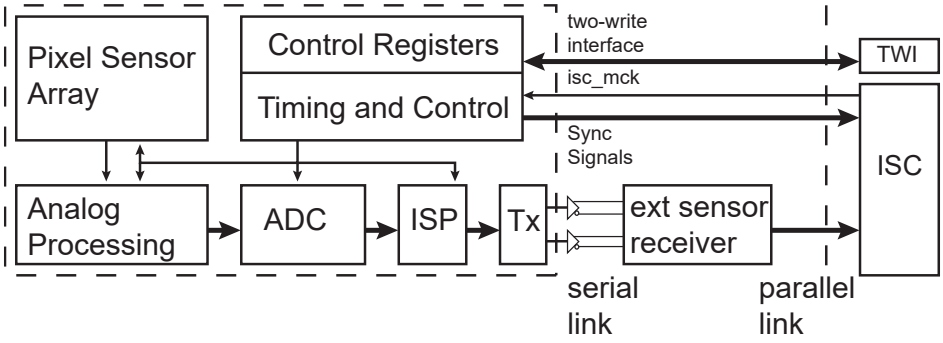



**Figure 51-8. Serial CMOS Sensor with External Parallel Bridge**




## 51.4 Product Dependencies

### 51.4.1 I/O Lines

The pins used for interfacing the ISC are multiplexed with the PIO lines. The programmer must first program the PIO controller to assign the ISC pins to their peripheral function. If I/O lines of the ISC are not used by the application, they can be used for other purposes by the PIO controller.

### 51.4.2 Power Management

The peripheral clock is not continuously provided to the ISC. The programmer must first enable the ISC clock in the Power Management Controller (PMC) before using the ISC.

### 51.4.3 Interrupt Sources

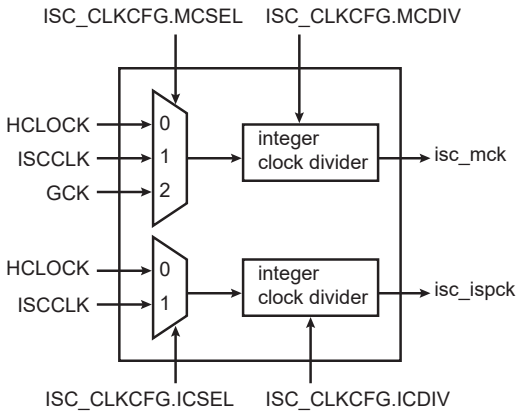
The ISC interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the ISC interrupt requires the Interrupt Controller to be programmed first.

## 51.5 Functional Description

### 51.5.1 ISC Clock Management

The ISC module provides the isc_mck output clock to the image sensor. The isc_mck clock has three selectable clock sources (ISC_CLKCFG.MCSEL) and one programmable clock divider (ISC_CLKCFG.MCDIV). The clock is enabled using the ISC_CLKEN.MCEN. The isc_mck is driven by the ISC and is the external reference clock of the CMOS sensor.

**Figure 51-9. Clock Divider Block Diagram**




The ISC digital pipeline requires internally a functional clock named isc_ispck. This clock is also fully programmable. This isc_ispck is enabled using the ISC_CLKEN.ICEN. This clock is mandatory for ISC operation. The ISC module is designed to accept input signals that are asynchronous to the isc_ispck. Synchronization is done internally as long as the following relationship holds:

isc_pck frequency is less than or equal to isc_ispck, and isc_ispck is greater than or equal to HCLOCK.

#### 51.5.1.1 Software Requirement

A software write operation to the ISC_CLKEN or ISC_CLKDIS register requires double clock domain synchronization and is not permitted when the ISC_CLKSR.SIP is asserted.

### 51.5.2 Parallel Interface Timing Description

The parallel interface protocol supports two operating modes.

**Figure 51-10. Free-Running Pixel Clock**

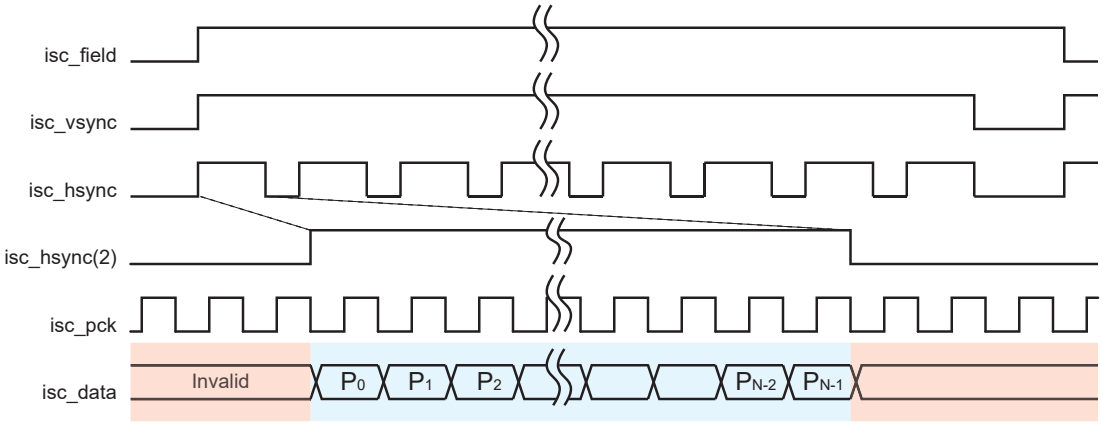
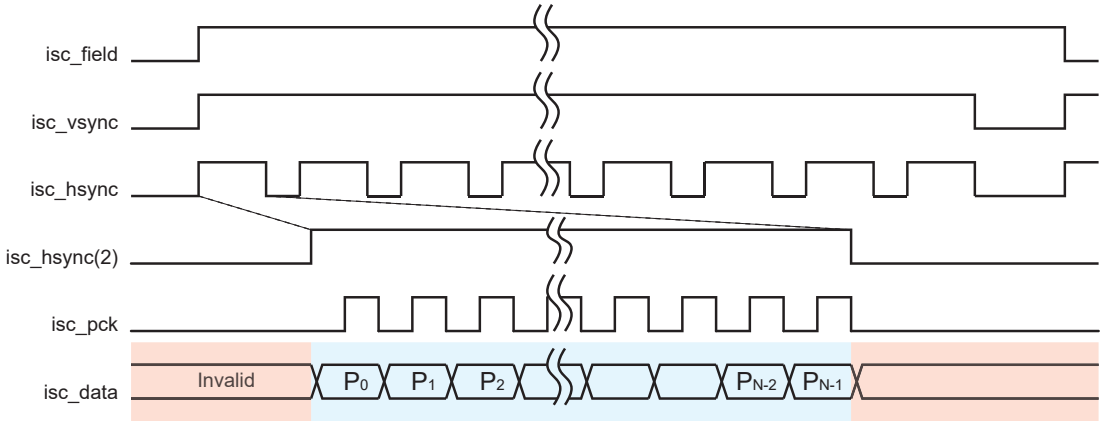


**Figure 51-11. Gated Pixel Clock**




### 51.5.3 BT.601/656/1120 Embedded Timing Synchronization Operation

The ISC module supports embedded synchronization decoding. When the ISC_PFE_CFG0.CCIR656 is set, the decoder is activated and signals isc_vsync isc_hsync are not used to decode the valid pixels. If the CCIR10_8N is set, the bitstream is 10 bits wide, otherwise it is only 8 bits wide. When the ISC_PFE_CFG0.CCIR_CRC is set, the decoder automatically corrects the error.

**Figure 51-12. Field/Segment Timing Relationship for Interlaced and Segmented Frame Systems**

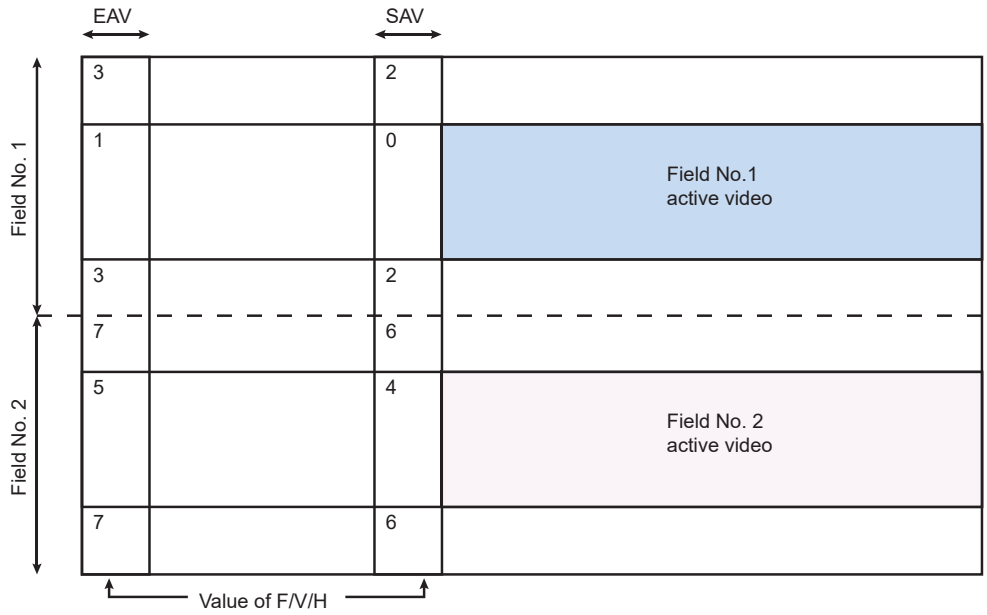
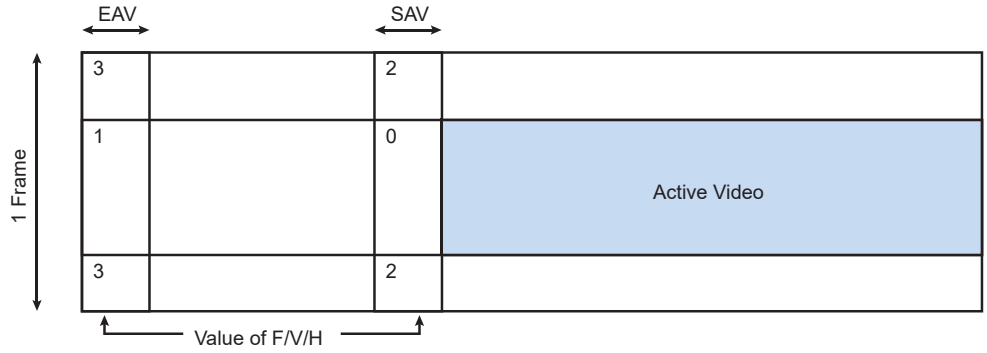


**Figure 51-13. Frame Timing Relationship for Progressive Systems**




### 51.5.4 Parallel Interface External Sensor Connections

#### 51.5.4.1 YCbCr, 10-bit CCIR656 with Embedded Synchronization

This mode is activated when ISC_PFE_CFG0.CCIR656 and ISC_PFE_CFG0.CCIR10_8N are both set.

| Interface Bit | First Word | Second Word | Third Word | Fourth Word |
|---|---|---|---|---|
| isc_data[11](MSB) | 1 | 0 | 0 | 1 |
| isc_data[10] | 1 | 0 | 0 | F |
| isc_data[9] | 1 | 0 | 0 | V |
| isc_data[8] | 1 | 0 | 0 | H |
| isc_data[7] | 1 | 0 | 0 | P3 |
| isc_data[6] | 1 | 0 | 0 | P2 |
| isc_data[5] | 1 | 0 | 0 | P1 |
| isc_data[4] | 1 | 0 | 0 | P0 |
| isc_data[3] | 1 | 0 | 0 | 0 |
| isc_data[2] | 1 | 0 | 0 | 0 |
| isc_data[1] | Not Used | Not Used | Not Used | Not Used |
| isc_data[0] | Not Used | Not Used | Not Used | Not Used |

#### 51.5.4.2 YCbCr, 8-bit CCIR656 with Embedded Synchronization

This mode is activated when ISC_PFE_CFG0.CCIR656 is set and ISC_PFE_CFG0.CCIR10_8N is cleared.

| Interface Bit | First Word | Second Word | Third Word | Fourth Word |
|---|---|---|---|---|
| isc_data[11](MSB) | 1 | 0 | 0 | 1 |
| isc_data[10] | 1 | 0 | 0 | F |
| isc_data[9] | 1 | 0 | 0 | V |
| isc_data[8] | 1 | 0 | 0 | H |
| isc_data[7] | 1 | 0 | 0 | P3 |

.....continued

| Interface Bit | First Word | Second Word | Third Word | Fourth Word |
|---|---|---|---|---|
| isc_data[6] | 1 | 0 | 0 | P2 |
| isc_data[5] | 1 | 0 | 0 | P1 |
| isc_data[4] | 1 | 0 | 0 | P0 |
| isc_data[3] | Not Used | Not Used | Not Used | Not Used |
| isc_data[2] | Not Used | Not Used | Not Used | Not Used |
| isc_data[1] | Not Used | Not Used | Not Used | Not Used |
| isc_data[0] | Not Used | Not Used | Not Used | Not Used |

#### 51.5.4.3 RAW Bayer Parallel Interface

The table below shows how to connect the data bus of a RAW Bayer sensor.

| Interface | Bayer 12-bit | Bayer 11-bit | Bayer 10-bit | Bayer 9-bit | Bayer 8-bit |
|---|---|---|---|---|---|
| isc_data[11](MSB) | DOUT[11] | DOUT[10] | DOUT[9] | DOUT[8] | DOUT[7] |
| isc_data[10] | DOUT[10] | DOUT[9] | DOUT[8] | DOUT[7] | DOUT[6] |
| isc_data[9] | DOUT[9] | DOUT[8] | DOUT[7] | DOUT[6] | DOUT[5] |
| isc_data[8] | DOUT[8] | DOUT[7] | DOUT[6] | DOUT[5] | DOUT[4] |
| isc_data[7] | DOUT[7] | DOUT[6] | DOUT[5] | DOUT[4] | DOUT[3] |
| isc_data[6] | DOUT[6] | DOUT[5] | DOUT[4] | DOUT[3] | DOUT[2] |
| isc_data[5] | DOUT[5] | DOUT[4] | DOUT[3] | DOUT[2] | DOUT[1] |
| isc_data[4] | DOUT[4] | DOUT[3] | DOUT[2] | DOUT[1] | DOUT[0] |
| isc_data[3] | DOUT[3] | DOUT[2] | DOUT[1] | DOUT[0] | Not Used |
| isc_data[2] | DOUT[2] | DOUT[1] | DOUT[0] | Not Used | Not Used |
| isc_data[1] | DOUT[1] | DOUT[0] | Not Used | Not Used | Not Used |
| isc_data[0] | DOUT[0] | Not Used | Not Used | Not Used | Not Used |

#### 51.5.4.4 Monochrome Parallel Interface

The table below shows how to connect the data bus of a Monochrome sensor.

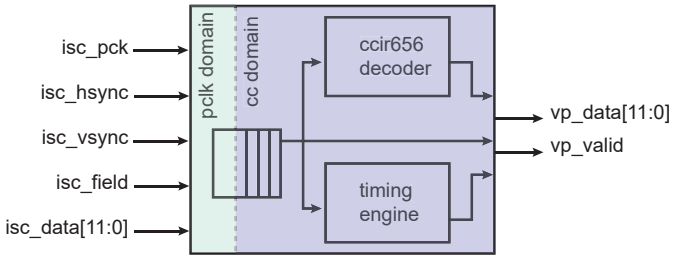
| Interface | Mono 12-bit | Mono 11-bit | Mono 10-bit | Mono 9-bit | Mono 8-bit |
|---|---|---|---|---|---|
| isc_data[11](MSB) | DOUT[11] | DOUT[10] | DOUT[9] | DOUT[8] | DOUT[7] |
| isc_data[10] | DOUT[10] | DOUT[9] | DOUT[8] | DOUT[7] | DOUT[6] |
| isc_data[9] | DOUT[9] | DOUT[8] | DOUT[7] | DOUT[6] | DOUT[5] |
| isc_data[8] | DOUT[8] | DOUT[7] | DOUT[6] | DOUT[5] | DOUT[4] |

.....continued

| Interface | Mono 12-bit | Mono 11-bit | Mono 10-bit | Mono 9-bit | Mono 8-bit |
|---|---|---|---|---|---|
| isc_data[7] | DOUT[7] | DOUT[6] | DOUT[5] | DOUT[4] | DOUT[3] |
| isc_data[6] | DOUT[6] | DOUT[5] | DOUT[4] | DOUT[3] | DOUT[2] |
| isc_data[5] | DOUT[5] | DOUT[4] | DOUT[3] | DOUT[2] | DOUT[1] |
| isc_data[4] | DOUT[4] | DOUT[3] | DOUT[2] | DOUT[1] | DOUT[0] |
| isc_data[3] | DOUT[3] | DOUT[2] | DOUT[1] | DOUT[0] | Not Used |
| isc_data[2] | DOUT[2] | DOUT[1] | DOUT[0] | Not Used | Not Used |
| isc_data[1] | DOUT[1] | DOUT[0] | Not Used | Not Used | Not Used |
| isc_data[0] | DOUT[0] | Not Used | Not Used | Not Used | Not Used |

### 51.5.5 Parallel Front End (PFE) Module

**Figure 51-14. PFE Block Diagram**




The Parallel Front End module performs data resampling across clock domain boundary. It includes a CCIR656 decoder used to convert a standard ITU-R BT.656 stream to 24-bit digital video. It also generates pixels, syncs flags and valid signals to the main video pipeline. It ouputs field, video and synchronization signals. The PFE can optionally crop and limit the incoming pixel stream to a predefined horizontal and vertical value. By default, the PFE only relies on the cmos sensor horizontal and vertical reference to sample the incoming pixel stream. A pixel is sampled if, and only if, the vertical and horizontal synchronizations are valid and a pixel clock edge is detected. ISC_PFE_CFG0.BPS shows the number of bits per sample. The PFE module outputs a 12-bit data on the vp_data[11:0] bus, and asserts the vp_valid signal when the data can be sampled.

| PFE VP_DATA Mapping | Raw Bayer 12-bit | Raw Bayer 10-bit | YUV422 8-bit | YUV422 10-bit | Mono 12-bit |
|---|---|---|---|---|---|
| VP_DATA[11] | RGGB[11] | RGGB[9] | YC422[7] | YC422[9] | Y[11] |
| VP_DATA[10] | RGGB[10] | RGGB[8] | YC422[6] | YC422[8] | Y[10] |
| VP_DATA[9] | RGGB[9] | RGGB[7] | YC422[5] | YC422[7] | Y[9] |
| VP_DATA[8] | RGGB[8] | RGGB[6] | YC422[4] | YC422[6] | Y[8] |
| VP_DATA[7] | RGGB[7] | RGGB[5] | YC422[3] | YC422[5] | Y[7] |
| VP_DATA[6] | RGGB[6] | RGGB[4] | YC422[2] | YC422[4] | Y[6] |
| VP_DATA[5] | RGGB[5] | RGGB[3] | YC422[1] | YC422[3] | Y[5] |

.....continued

| PFE VP_DATA Mapping | Raw Bayer 12-bit | Raw Bayer 10-bit | YUV422 8-bit | YUV422 10-bit | Mono 12-bit |
|---|---|---|---|---|---|
| VP_DATA[4] | RGGB[4] | RGGB[2] | YC422[0] | YC422[2] | Y[4] |
| VP_DATA[3] | RGGB[3] | RGGB[1] | YC422[7] or 0 | YC422[1] | Y[3] |
| VP_DATA[2] | RGGB[2] | RGGB[0] | YC422[6] or 0 | YC422[0] | Y[2] |
| VP_DATA[1] | RGGB[1] | RGGB[9] or 0 | YC422[5] or 0 | YC422[9] or 0 | Y[1] |
| VP_DATA[0] | RGGB[0] | RGGB[8] or 0 | YC422[4] or 0 | YC422[8] or 0 | Y[0] |

**Note:** When ISC_PFE_CFG0.REP is set, missing VP_DATA LSBs are replaced with replicated LSBs of the incoming stream, otherwise they are forced to zero.

The PFE module also includes logic to synchronize capture request with the incoming pixel stream. Two operating modes are available: Single Shot and Continuous Acquisition. When the ISC_PFE_CFG0.CONT is cleared, the ISC transfers a single image to memory,

**Figure 51-15. Single Shot Mode**

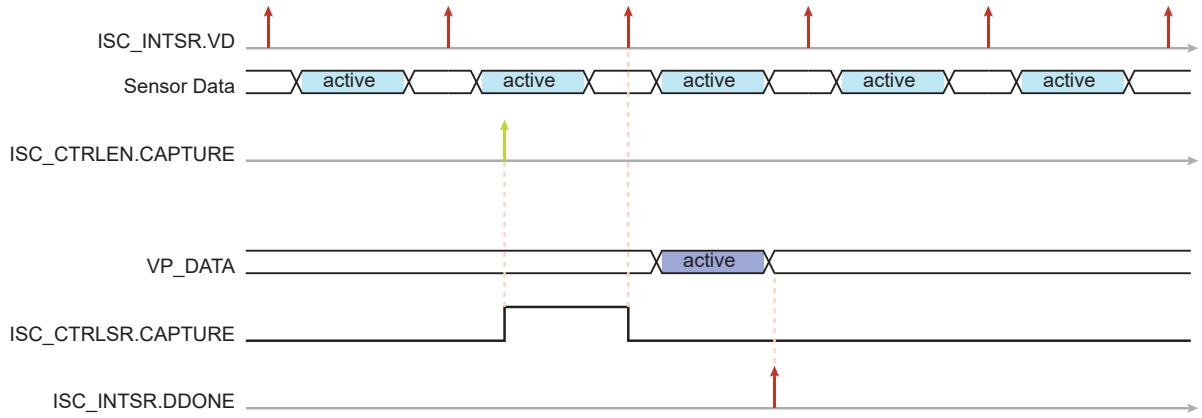
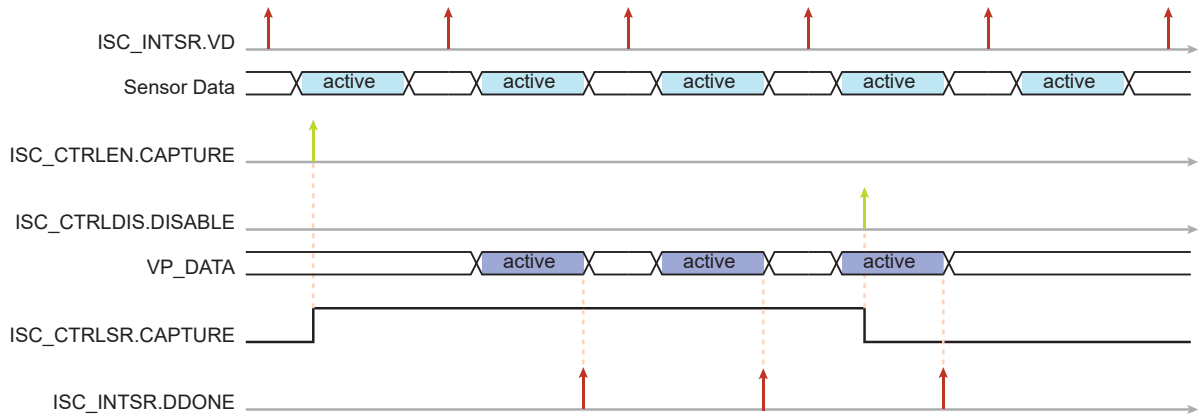
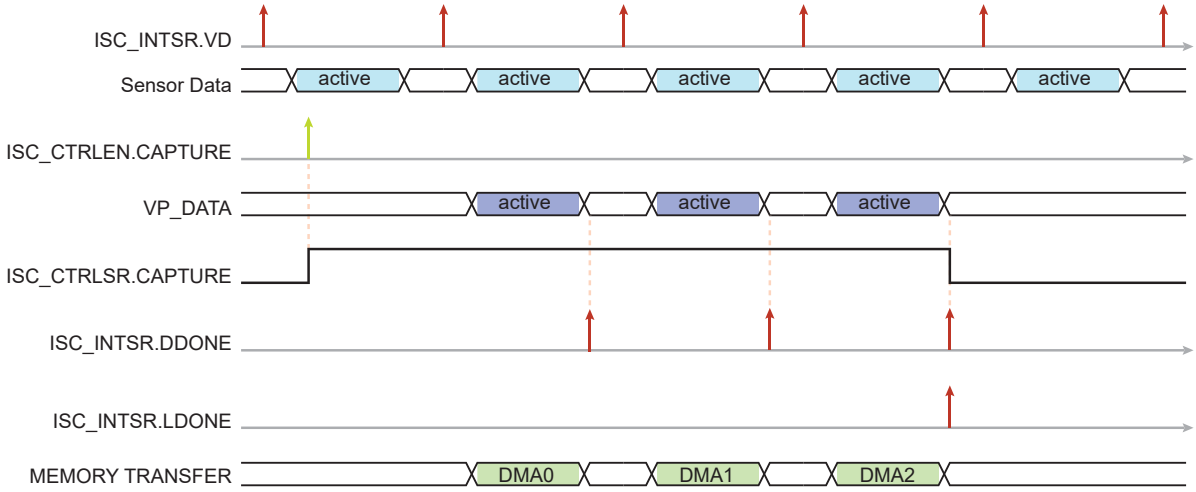



**Figure 51-16. Continuous Acquisition Mode**




When Continuous Acquisition mode is activated (ISC_PFE_CFG0.CONT is set), the data transfer terminates when either a DMA end of list is reached, a software disable is performed or a software reset is activated. The ISC_INTSR.DDONE is set at the end of the DMA data transfer.

**Figure 51-17. Continuous Acquisition, DMA Terminated**



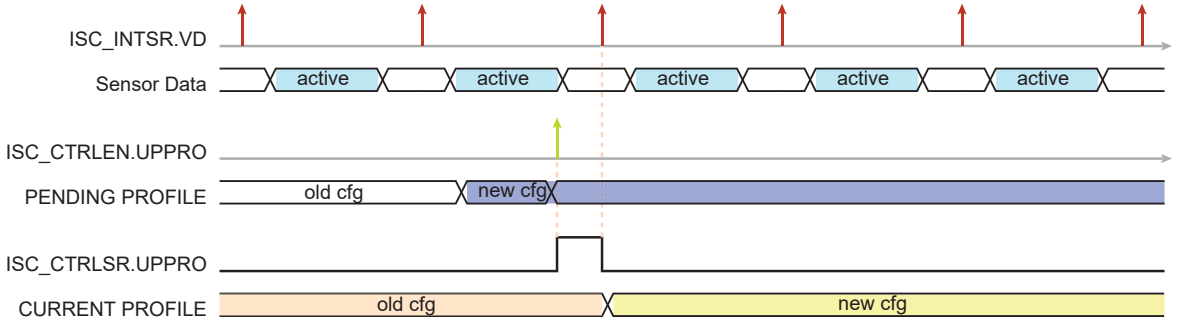

The linked list DMA transfer is terminated when an item of the list is programmed with ISC_DCTRL.DE cleared or when ISC_DNDA.NDA is equal to zero. This configuration also clears ISC_CTRLSR.CAPTURE and sets the ISC_INTSR.LDONE interrupt flag.

The linked list DMA transfer starts if ISC_DCTRL.DE is set and if ISC_DNDA.NDA is different from zero.

#### 51.5.5.1 Update the ISC Profile

Each ISC register is double-buffered to simplify the software configuration and the synchronization with the associated frame buffer. When the configuration of the ISC is modified, ISC_CTRLEN.UPPRO must be set to transfer the configuration from the input buffer to the ISC video pipeline.

**Figure 51-18. Update Profile Timing Diagram**



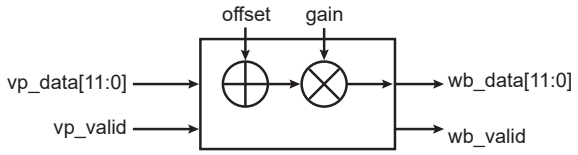

#### 51.5.5.2 Software Requirements

Writing to the ISC_CTRLEN or ISC_CTRLDIS register requires a double domain synchronization, so it is forbidden to write these registers when ISC_CTRLSR.SIP is asserted.

### 51.5.6 White Balance (WB) Module

The White Balance (WB) module captures the vp_data[11:0] bus from the PFE module when the vp_valid signal is asserted, and it generates a wb_data[11:0] data along with its validity signal wb_valid. When ISC_WB_CTRL.ENABLE is set, each Bayer color component (R, Gr, B, Gb) can be manually adjusted using an offset and a gain. The Bayer pattern is adjustable using ISC_WB_CFG.BAYCFG.

**Figure 51-19. WB Block Diagram**



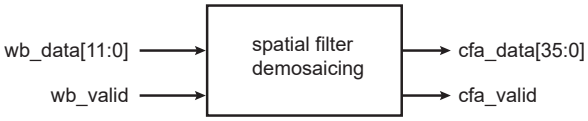

There are four {gain, offset} sets for each Bayer component. The output value is clipped.

| ISC_WB_CTRL.ENABLE | WB_DATA Slice | Value |
|---|---|---|
| 0 | wb_data[11:0] | vp_data[11:0] |
| 1 | wb_data[11:0] | clipped((vp_data[11:0]+offset)*gain) |

### 51.5.7 Color Filter Array (CFA) Interpolation Module

In a single-sensor system, each cell on the sensor has a specific color filter and microlens positioned above it. The raw data obtained from the sensor do not have the full R/G/B information at each cell position. Color interpolation is required to retrieve the missing components. The CFA module samples the wb_data[11:0] 12-bit bus when wb_valid is asserted and generates a 36-bit width data bus cfa_data[35:0] with the validity bit cfa_valid.

**Figure 51-20. CFA Block Diagram**




| ISC_CFA_CTRL.ENABLE | CFA_DATA Slice | Value |
|---|---|---|
| 0 | cfa_data[35:24] | wb_data[11:0] |
| | cfa_data[23:12] | wb_data[11:0] |
| | cfa_data[11:0] | wb_data[11:0] |
| 1 | cfa_data[35:24] | R = spatial_filter_R(wb_data[11:0]) |
| | cfa_data[23:12] | G = spatial_filter_G(wb_data[11:0]) |
| | cfa_data[11:0] | B = spatial_filter_B(wb_data[11:0]) |

The filter kernel size is 5, and requires two additional lines to initialize the filter. When ISC_CFA_CFG.EITPOL is set, the missing information is interpolated from the nearest neighbor. If ISC_CFA_CFG.EITPOL is cleared, only valid pixels are used to initialize the filter kernel, but the output number of lines is less than the input number of lines. In that case, four lines are consumed to fill the kernel.

#### 51.5.7.1 Frame Size Requirement when Edge Interpolation is Off, ISC_CFA_CFG.EITPOL Cleared

- Minimum number of rows (in): 5
- Minimum number of columns (in): 5
- Number or rows after CFA: Number of rows (in) - 4

- Number of columns after CFA: Number of columns (in) - 4

#### 51.5.7.2 Frame Size Requirement when Edge Interpolation is On, ISC_CFA_CFG.EITPOL Set

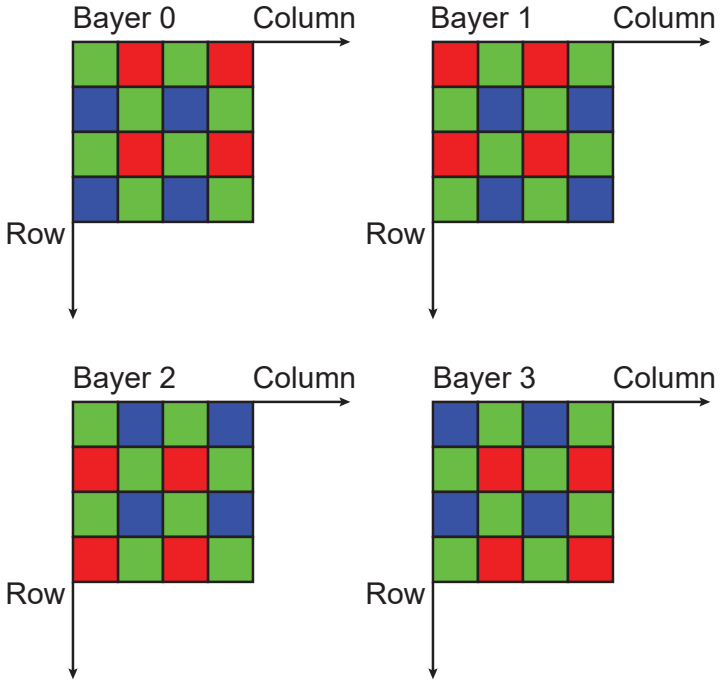
- Minimum number of rows (in): 3
- Minimum number of columns (in): 3
- Number of rows after CFA: Number of rows (in)
- Number of columns after CFA: Number of columns (in)

#### 51.5.7.3 Bayer Mode and Edge Interpolation Description

When Edge Interpolation mode (ISC_CFA_CFG.EITPOL) is activated, dummy lines are generated using rows and columns replication.

The CFA module supports four sensor alignments using ISC_CFA_CFG.BAYCFG. See the figure below.

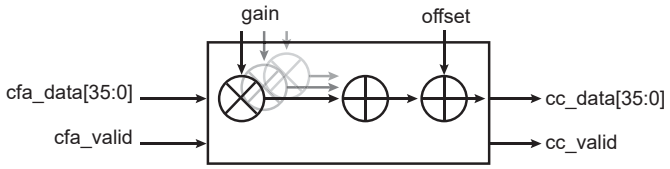
**Figure 51-21. Supported Color Filter Array Patterns**




### 51.5.8 Color Correction (CC) Module

RGB color correction is used to compensate for cross color bleeding in the filter used with the image sensor. The module samples the cfa_data[35:0] 36-bit bus when cfa_valid is asserted and generate a cc_data[35:0] 36-bit wide bus and a cc_valid signal.

**Figure 51-22. CC Block Diagram**




There are three {gain, offset} sets for color component R, G, B.

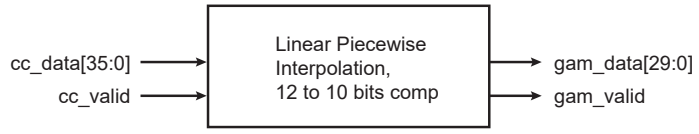
| ISC_CC_CTRL.ENABLE | CC_DATA Slice | Value |
|---|---|---|
| 0 | cc_data[35:24] | cfa_data[11:0] |
| | cc_data[23:12] | cfa_data[11:0] |
| | cc_data[11:0] | cfa_data[11:0] |
| 1 | cc_data[35:24] | R=clipped(sum(cfa_data_x * gain_Rx) + offset_R) |
| | cc_data[23:12] | G=clipped(sum(cfa_data_x * gain_gx) + offset_g) |
| | cc_data[11:0] | B=clipped(sum(cfa_data_x * gain_Bx) + offset_B) |

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} \text{RRGAIN} & \text{RGGAIN} & \text{RBGAIN} \\ \text{GRGAIN} & \text{GGGAIN} & \text{GBGAIN} \\ \text{BRGAIN} & \text{BGGAIN} & \text{BBGAIN} \end{bmatrix} \times \begin{bmatrix} \text{cfa_data}[35:24] \\ \text{cfa_data}[23:12] \\ \text{cfa_data}[11:0] \end{bmatrix} + \begin{bmatrix} \text{ROFST} \\ \text{GOFST} \\ \text{BOFST} \end{bmatrix}$$

### 51.5.9 Gamma Curve (GAM) Module

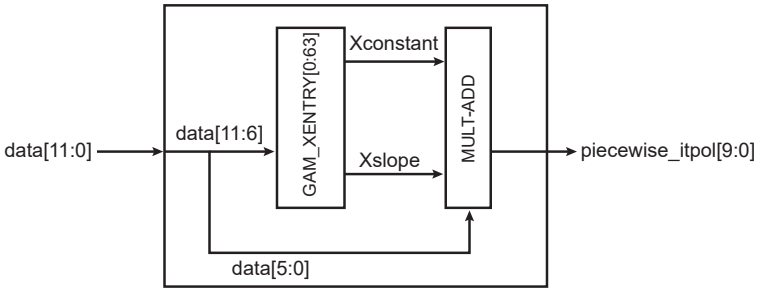
The GAM module samples the cc_data[35:0] bus when cc_valid is asserted, and generates gam_data[29:0] 30-bit width data along with the validity signal gam_valid. Imaging devices have non-linear characteristics, but the transfer function is approximated by a power function. The intensity of each of the linear RGB components is transformed to a non-linear signal through the use of the gamma correction submodule. The power function is linearly interpolated using 64 breakpoints. This also performs a 12-bit to 10-bit compression. The polynomial for the linear interpolation between breakpoints is i and i +1. Consequently, for each breakpoint, two values are required: constant and slope. The table values are programmable through the user interface when the gamma correction module is disabled (ISC_GAM_CTRL.ENABLE is cleared). ISC_GAM_RENTRY is used for Red gamma correction. ISC_GAM_GENTRY is used for Green gamma correction. ISC_GAM_BENTRY is used for Blue gamma correction. Each table entry is composed of a 10-bit (signed) slope and a 10-bit constant.

**Figure 51-23. GAM Block Diagram**




| ISC_GAM_CTRL.ENABLE | ISC_GAM_CTRL.XLUT | GAM_DATA Slice | Value |
|---|---|---|---|
| 0 | 0 | gam_data[29:0] | cc_data[29:0] |
| 1 | 0 | gam_data[29:20] | cc_data[35:26] |
| | | gam_data[19:10] | cc_data[23:14] |
| | | gam_data[9:0] | cc_data[11:2] |
| 1 | 1 | gam_data[29:20] | R=piecewise_itpol(cc_data_r[35:24]) |
| | | gam_data[19:10] | G=piecewise_itpol(cc_data_r[23:12]) |
| | | gam_data[9:0] | B=piecewise_itpol(cc_data_r[11:0]) |

**Figure 51-24. Piecewise Linear Interpolation Block Diagram**

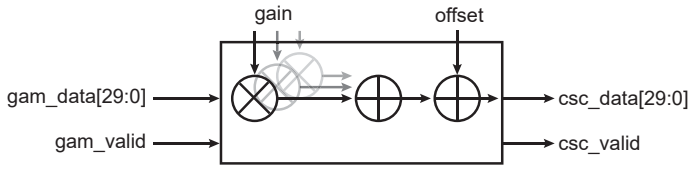



The interpolation consists of three tables that store the function values GAM_XENTRY[0:63] where X stands for R, G and B. The input of the table has six bits. It outputs a slope and a constant. The slope is later multiplied by the data lsb (6-bit) and added to a constant. The final value is the gamma-corrected value of the input. This module performs a 12-to-10 compression.

### 51.5.10 Color Space Conversion (CSC) Module

By converting an image from RGB to YCbCr color space, it is possible to separate Y, Cb and Cr information. The CSC samples the gam_data[29:0] 30-bit data bus, extracts YCbCr information from the sampled data, and then generates the color-converted data csc_data[29:0] and the validity signal csc_valid.

**Figure 51-25. CSC Block Diagram**



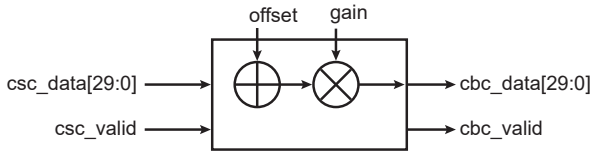

| ISC_CSC_CTRL.ENABLE | CSC_DATA Slice | Value |
|---|---|---|
| 0 | csc_data[29:0] | gam_data[29:0] |
| 1 | csc_data[29:20] | Y = clipped(sum(gam_data_x * gain_Yx) + offset_y << 2) |
| | csc_data[19:10] | Cb = clipped(sum(gam_data_x * gain_Cbx)+offset_cb << 2) |
| | csc_data[9:0] | Cr = clipped(sum(gam_data_x * gain_Crx)+offset_cr << 2) |

$$\begin{bmatrix} Y \\ CB \\ CR \end{bmatrix} = \begin{bmatrix} YR & YG & YB \\ CBR & CBG & CBB \\ CRR & CRG & CRB \end{bmatrix} \times \begin{bmatrix} gam_data[29:20] \\ gam_data[19:10] \\ gam_data[9:0] \end{bmatrix} + \begin{bmatrix} YOFST \\ CBOFST \\ CROFST \end{bmatrix}$$

### 51.5.11 Contrast and Brightness

Luminance is adjusted through the use of Brightness Offset and Contrast Gain. Chrominance is left unchanged. The CBC samples the csc_data[29:0] 30-bit bus when csc_valid is asserted and generates cbc_data[29:0] with the validity signal cbc_valid.

**Figure 51-26. CBC Block Diagram**

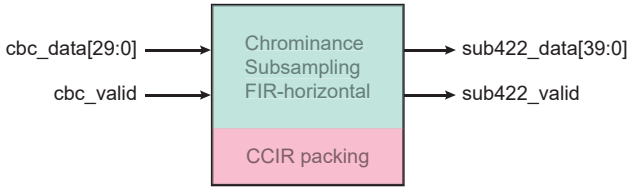



| ISC_CBC_CTRL.ENABLE | ISC_CBC_CFG.CCIR | CBC_DATA Slice | Value |
|---|---|---|---|
| 0 | 0 | cbc_data[29:0] | csc_data[29:0] |
| 1 | 0 | cbc_data[29:20] | Y = clipped((csc_data[29:20]+offset)*gain) |
| | | cbc_data[19:10] | Cb = csc_data[19:10] |
| | | cbc_data[9:0] | Cr = csc_data[9:0] |
| 1 | 1 | cbc_data[29:10] | 0 |
| | | cbc_data[9:0] | ccir656 stream with luminance correction |

### 51.5.12 4:4:4 To 4:2:2 Chrominance Horizontal Subsampler (SUB422) Module

The color space conversion output stream is a full-bandwidth YCbCr 4:4:4 signal. The chrominance subsampling divides the horizontal chrominance sampling rate by two. A horizontal low pass filter is applied to avoid aliasing effect. The SUB422 module samples 444 full scale YCbCr cbc_data[29:0] 30-bit data, performs horizontal subsampling and generates the sub422_data[39:0] 40-bit data bus with its validity signal sub422_valid.

**Figure 51-27. SUB422 Block Diagram**




| ISC_SUB422_CTRL.ENABLE | ISC_SUB422_CFG.CCIR | SUB422_DATA Slice | Value |
|---|---|---|---|
| 0 | 0 | sub422_data[29:0] | cbc_data[29:0] |
| 1 | 0 | sub422_data[39:30] | Y1= cbc_data1[29:20] |
| | | sub422_data[29:20] | Y0 = cbc_data0[29:20] |
| | | sub422_data[19:10] | Cb = filter_hor(cbc_data[19:10]) |
| | | sub422_data[9:0] | Cr = filter_hor(cbc_data[9:0]) |

| .....continued | | | |
|---|---|---|---|
| **ISC_SUB422_CTRL.ENABLE** | **ISC_SUB422_CFG.CCIR** | **SUB422_DATA Slice** | **Value** |
| 1 | 1 | sub422_data[39:30] | Y1 = cbc_data[9:0] |
| | | sub422_data[29:20] | Y0 = cbc_data[9:0] |
| | | sub422_data[19:10] | Cb = cbc_data[9:0] |
| | | sub422_data[9:0] | Cr = cbc_data[9:0] |

The filter_hor function included in the sub422 module is the chrominance horizontal filter.

| sub422 data slice | YCbCr mapping |
|---|---|
| sub422_data[39:30] | Y1 (sample n) |
| sub422_data[29:20] | Y0 (sample n-1) |
| sub422_data[19:10] | Cb (from filter) |
| sub422_data[9:0] | Cr (from filter) |

The filter chrominance position is selectable through the use of ISC_SUB422_CFG.FILTER.

**Figure 51-28. Cosited Filter Configuration**

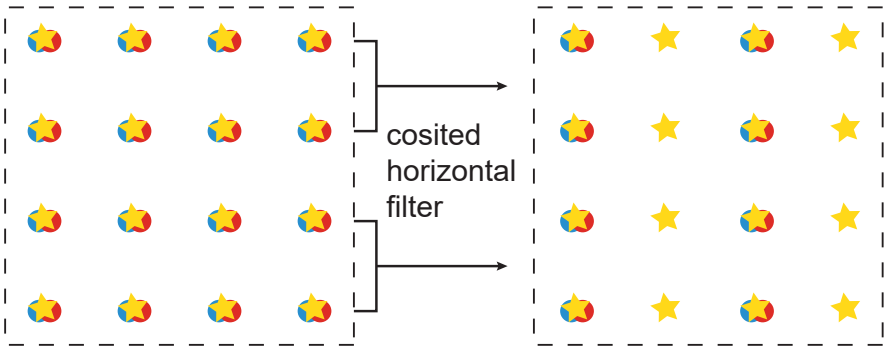
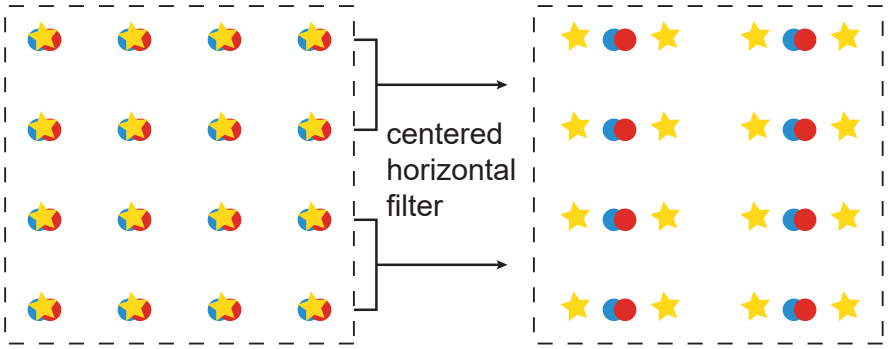



**Figure 51-29. Centered Filter Configuration**



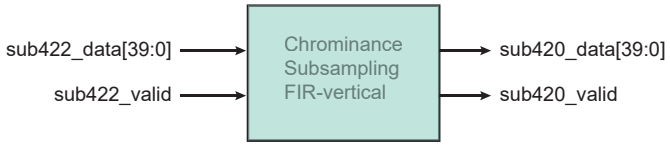

The SUB422 module performs luminance and chrominance packing. When the line length is odd, the missing luminance is a copy of the last but one luminance. It also means that the final dma stream written to memory is equal to the original horizontal size plus one when the line length is odd.

| SUB422_DATA Slice | Line Length Even | Line Length Odd |
|---|---|---|
| sub422_data[39:30] | Y(n) | Y(n-1) |
| sub422_data[29:20] | Y(n-1) | Y(n-1) |
| sub422_data[19:10] | Cb (filtered) | Cb (filtered) |
| sub422_data[9:0] | Cr (filtered) | Cr (filtered) |

### 51.5.13 4:2:2 To 4:2:0 Chrominance Vertical Subsampler (SUB420) Module

The chrominance subsampling divides the vertical chrominance sampling rate by two. A vertical low pass filter is applied to avoid aliasing effect. Two different filters are used when the source frame is interlaced, and the filter configuration depends on the field value (the field is propagated in the video pipeline).

**Figure 51-30. SUB420 Block Diagram**




The SUB420 module samples the sub422_data[39:0] 40-bit data when sub422_valid is asserted, then it performs a vertical subsampling and generates a valid sub420_data[39:0] 40-bit word and the corresponding sub420_valid signal.

| ISC_CFA_CTRL.ENABLE | SUB420_DATA Slice | Value |
|---|---|---|
| 0 | sub420_data[39:0] | sub422_data[39:0] |
| 1 | sub420_data[39:30] | Y1 = sub422_data[39:30] |
| | sub420_data[29:20] | Y0 = sub422_data[29:20] |
| | sub420_data[19:10] | Cb = filter_ver(sub422[19:10]) |
| | sub420_data[9:0] | Cr = filter_ver(sub422[9:0]) |

The vertical filter is a two-tap filter; for progressive content the coefficient i {1, 1}. When an interlaced field is downsampled, the coefficients are different between the top and the bottom fields.

**Figure 51-31. Vertical Chrominance Filter for Progressive Content (Cosited Chrominance Example)**

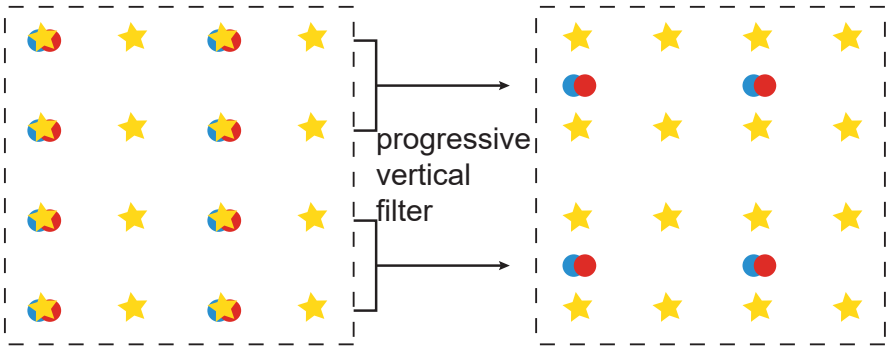


**Figure 51-32. Field-dependent Chrominance Filter for Interlaced Content (Cosited Chrominance Example)**

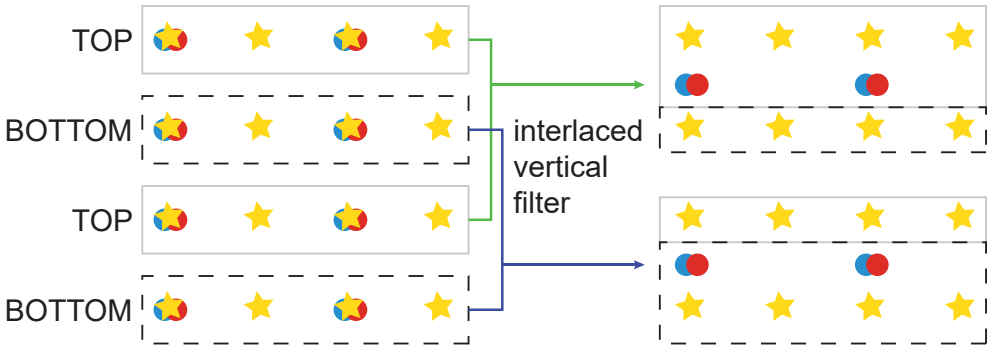



**Table 51-1. Filter Configuration**

| ISC_SUB420_CTRL.FILTER | Field | Filter Configuration |
|---|---|---|
| 0 | progressive | {1, 1} |
| 1 | 0 (TOP) | {3, 1} |
| | 1 (BOTTOM) | {1, 3} |

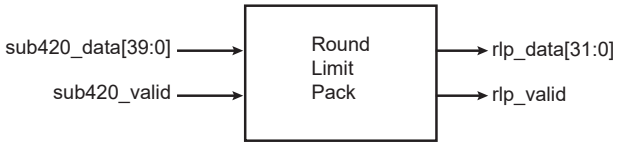
**Table 51-2. Output Line Length Configuration**

| SUB420 Input Number of Rows | SUB420 Luminance Rows | SUB420 Chrominance Rows |
|---|---|---|
| M rows, M odd | M rows | (M+1)/2 rows |
| M rows, M even | M rows | M/2 rows |

### 51.5.14 Rounding, Limiting and Packing (RLP) Module

This module is used to round, limit and pack in the incoming pixel stream before the DMA master module. The RLP samples the sub420_data[39:0] 40-bit data bus and generates rlp_data[31:0] 32-bit data words with the associated validity signal rlp_valid.

**Figure 51-33. RLP Block Diagram**




| ISC_RLP_CFG | RLP_DATA Slice | Value |
|---|---|---|
| DAT8 | rlp_data[31:8] | 0 |
| | rlp_data[7:0] | sub420_data[11:4] |
| DAT9 | rlp_data[31:9] | 0 |
| | rlp_data[8:0] | sub420_data[11:3] |
| DAT10 | rlp_data[31:10] | 0 |
| | rlp_data[9:0] | sub420_data[11:2] |

.....continued

| ISC_RLP_CFG | RLP_DATA Slice | Value |
|---|---|---|
| DAT11 | rlp_data[31:11] | 0 |
| | rlp_data[10:0] | sub420_data[11:1] |
| DAT12 | rlp_data[31:12] | 0 |
| | rlp_data[11:0] | sub420_data[11:0] |
| DATY8 | rlp_data[31:8] | 0 |
| | rlp_data[7:0] | Y = rounded(sub420_data[29:22]) |
| DATY10 | rlp_data[31:8] | 0 |
| | rlp_data[7:0] | Y = sub420_data[29:20]) |
| ARGB444 | rlp_data[31:16] | 0 |
| | rlp_data[15:12] | A = alpha[7:4] |
| | rlp_data[11:8] | R = sub420_data[29:26] |
| | rlp_data[7:4] | G = sub420_data[19:16] |
| | rlp_data[3:0] | B = sub420_data[9:6] |
| ARGB555 | rlp_data[31:16] | 0 |
| | rlp_data[15] | A = alpha[7] |
| | rlp_data[14:10] | R = sub420_data[29:25] |
| | rlp_data[9:5] | G = sub420_data[19:15] |
| | rlp_data[4:0] | B = sub420_data[9:5] |
| RGB565 | rlp_data[31:16] | 0 |
| | rlp_data[15:11] | R = sub420_data[29:25] |
| | rlp_data[10:5] | G = sub420_data[19:14] |
| | rlp_data[4:0] | B = sub420_data[9:5] |
| RGB32 | rlp_data[31:24] | A = alpha[7:0] |
| | rlp_data[23:16] | R = sub420_data[29:22] |
| | rlp_data[15:8] | G = sub420_data[19:12] |
| | rlp_data[7:0] | B = sub420_data[9:2] |
| YCbCr422, YCbCr420 | rlp_data[31:24] | Y1 = round(sub420_data[39:32]) |
| | rlp_data[23:16] | Y0 = round(sub420_data[29:22]) |
| | rlp_data[15:8] | Cb = round(sub420_data[19:12]) |
| | rlp_data[7:0] | Cr = round(sub420_data[9:2]) |

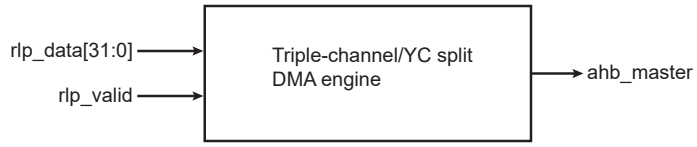
| .....continued | | |
|---|---|---|
| **ISC_RLP_CFG** | **RLP_DATA Slice** | **Value** |
| YCbCr422, YCbCr420 | rlp_data[31:24] | Y1 = round_limit(sub420_data[39:32]) |
| | rlp_data[23:16] | Y0 = round_limit(sub420_data[29,22]) |
| | rlp_data[15:8] | Cb = round_limit(sub420_data[19,12]) |
| | rlp_data[7:0] | Cr = round_limit(sub420_data[9:2]) |
| Undefined | rlp_data[31:0] | sub420_data[31:0] |

| ISC_RLP_CFG | 8-bit Full Range | 8-bit Limited Range |
|---|---|---|
| Y | 0–255 | 16–235 |
| Cb | 0–255 | 16–240 |
| Cr | 0–255 | 16–240 |

### 51.5.15 DMA Interface

The descriptor-based DMA interface supports multiple buffers. A DMA stride value shows the offset between two consecutive lines (in bytes). If the stride is set to zero, the frame buffer is contiguous. When ISC_DCTRL.WB is set (Write Back), the DMA interface performs a single write operation to the ISC_DCTRL register, and sets ISC_DCTRL[7] to one and ISC_DCTRL[6] to the value of the frame field when interlaced content is being used. That means that interlaced fields are tagged with their relevant field values. The Write Back operation is always performed when the whole frame has been transferred to memory.

**Figure 51-34. DMA Engine Block Diagram**




| ISC_DCFG.IMODE | DMA Engine Input Data |
|---|---|
| PACKED8 | rlp_data[7:0] |
| PACKED16 | rlp_data[15:0] |
| PACKED32 | rlp_data[31:0] |
| YC422SP | rlp_data[31:0] |
| YC422P | rlp_data[31:0] |
| YC420SP | rlp_data[31:0] |
| YC420P | rlp_data[31:0] |

When a bus error is detected, an interrupt flag is set. If the error occurs on a write operation, ISC_INTSR.WERR is asserted. If the error occurs on a read operation, ISC_INTSR.RERR is asserted. ISC_INTSR.WERRID gives details on the first error channel identifier.

#### 51.5.15.1 Descriptor Memory Address Mapping

| ISC_DCFG.IMODE | ISC_DAD0.AD0 | ISC_DAD1.AD1 | ISC_DAD2.AD2 |
|---|---|---|---|
| PACKED8, PACKED16, PACKED32 | data address | not used | not used |
| YC422SP | Y address | CbCr address | not used |
| YC422P | Y address | Cb address | Cr address |
| YC420SP | Y address | CbCr address | not used |
| YC420P | Y address | Cb address | Cr address |

#### 51.5.15.2 Descriptor Memory Mapping

Three descriptor views are available. Descriptor view 0 is used when the pixel or data stream is packed. Descriptor view 1 is used for YCbCr semi-planar pixel stream. Descriptor view 2 is used for YCbCr planar pixel stream.

**Table 51-3. ISC_DCTRL.DVIEW = 0**

| Address | Register |
|---|---|
| ISC_DNDA+0x00 | ISC_DCTRL |
| ISC_DNDA+0x04 | ISC_DNDA |
| ISC_DNDA+0x08 | ISC_DAD0 |
| ISC_DNDA+0x0C | ISC_DST0 |

**Table 51-4. ISC_DCTRL.DVIEW = 1**

| Address | Register |
|---|---|
| ISC_DNDA+0x00 | ISC_DCTRL |
| ISC_DNDA+0x04 | ISC_DNDA |
| ISC_DNDA+0x08 | ISC_DAD0 |
| ISC_DNDA+0x0C | ISC_DST0 |
| ISC_DNDA+0x10 | ISC_DAD1 |
| ISC_DNDA+0x14 | ISC_DST1 |

**Table 51-5. ISC_DCTRL.DVIEW = 2**

| Address | Register |
|---|---|
| ISC_DNDA+0x00 | ISC_DCTRL |
| ISC_DNDA+0x04 | ISC_DNDA |
| ISC_DNDA+0x08 | ISC_DAD0 |
| ISC_DNDA+0x0C | ISC_DST0 |
| ISC_DNDA+0x10 | ISC_DAD1 |
| ISC_DNDA+0x14 | ISC_DST1 |

| .....continued | |
|---|---|
| **Address** | **Register** |
| ISC_DNDA+0x18 | ISC_DAD2 |
| ISC_DNDA+0x1C | ISC_DST2 |

#### 51.5.15.3 Example: Memory Mapping for 16-bit Packed, DMA Interface IMODE = 1 at ISC_DAD0.AD0 Location

**Table 51-6. DAT8 Packing (ISC_RLP_CFG.MODE)**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RAW12 | – | – | – | – | – | – | – | – | rlp_data1[7:0] | | | | | | | | – | – | – | – | – | – | – | – | rlp_data0[7:0] | | | | | | | |

**Table 51-7. DAT9 Packing (ISC_RLP_CFG.MODE)**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RAW12 | – | – | – | – | – | – | – | rlp_data1[8:0] | | | | | | | | | – | – | – | – | – | – | – | rlp_data0[8:0] | | | | | | | | |

**Table 51-8. DAT10 Packing (ISC_RLP_CFG.MODE)**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RAW12 | – | – | – | – | – | – | rlp_data1[9:0] | | | | | | | | | | – | – | – | – | – | – | rlp_data0[9:0] | | | | | | | | | |

**Table 51-9. DAT11 Packing (ISC_RLP_CFG.MODE)**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RAW12 | – | – | – | – | – | rlp_data1[10:0] | | | | | | | | | | | – | – | – | – | – | isc_data0[10:0] | | | | | | | | | | |

**Table 51-10. DAT12 Packing (ISC_RLP_CFG.MODE)**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RAW12 | – | – | – | – | rlp_data1[11:0] | | | | | | | | | | | | – | – | – | – | rlp_data0[11:0] | | | | | | | | | | | |

#### 51.5.15.4 Example: Memory Mapping for 12-bit YC420SP, DMA Interface IMODE = 5

**Table 51-11. Y Channel Located at ISC_DAD0.AD0 Memory Address**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Y 8-bit | rlp_data1[31:24] | | | | | | | | rlp_data1[23:16] | | | | | | | | rlp_data0[31:24] | | | | | | | | rlp_data0[23:16] | | | | | | | |

**Table 51-12. CbCr Channel Located at ISC_DAD1.AD1 Memory Address**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CC 16-bit | rlp_data1[15:0] | | | | | | | | | | | | | | | | rlp_data0[15:0] | | | | | | | | | | | | | | | |

#### 51.5.15.5 Example: Memory Mapping for 12-bit YC420P, DMA Interface IMODE = 6

**Table 51-13. Y Channel Located at ISC_DAD0.AD0 Memory Address**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Y 8-bit | rlp_data1[31:24] | | | | | | | | rlp_data1[23:16] | | | | | | | | rlp_data0[31:24] | | | | | | | | rlp_data0[23:16] | | | | | | | |

**Table 51-14. Cb Channel Located at ISC_DAD1.AD1 Memory Address**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Cb 8-bit | rlp_data3[15:8] | | | | | | | | rlp_data2[15:8] | | | | | | | | rlp_data1[15:8] | | | | | | | | rlp_data0[15:8] | | | | | | | |

**Table 51-15. Cr Channel Located at ISC_DAD2.AD2**

| Mem addr | 0x3 | | | | | | | | 0x2 | | | | | | | | 0x1 | | | | | | | | 0x0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Cr 8-bit | rlp_data3[7:0] | | | | | | | | rlp_data2[7:0] | | | | | | | | rlp_data1[7:0] | | | | | | | | rlp_data0[7:0] | | | | | | | |

### 51.5.16 Histogram Module

For each possible pixel value, the histogram counts the number of times the value was encountered in the current image. RGGB Bayer, RAW data or luminance histogram are available. There are 512 entries

in the histogram entries, and each histogram bin can count up to 2^20 data. As the table entries are limited, each bin is actually a range, i.e., least significant bits are ignored. A write to ISC_CTRLEN.HISREQ initiates a new histogram. The counting operation ends when ISC_INTSR.HISDONE is set. At that time, a software or hardware dma transfer copies the table from the interface to the internal or external memory. To clear the table content (for a new operation), use ISC_CTRLEN.HISCLR. An automatic clear (reset after read) is available when ISC_HIS_CFG.RAR is set. In that case, as soon as the data is read from the table, the table entry is cleared.

**Figure 51-35. Histogram Block Diagram**

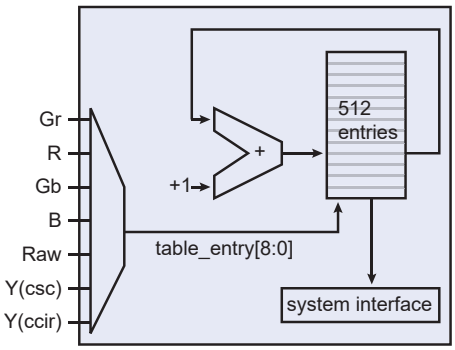
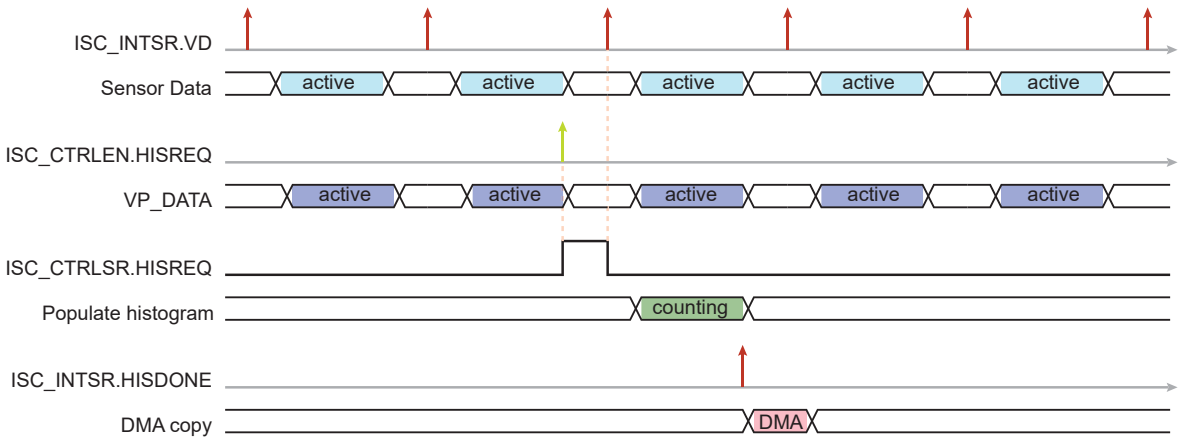



**Figure 51-36. Histogram Request Timing Diagram**



## 51.6 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | ISC_CTRLEN | 7:0 | | | | | HISCLR | HISREQ | UPPRO | CAPTURE |
| | | 15:8 | | | | | | | FUPPRO | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | ISC_CTRLDIS | 7:0 | | | | | | | | DISABLE |
| | | 15:8 | | | | | | | | SWRST |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | ISC_CTRLSR | 7:0 | | | | FIELD | | HISREQ | UPPRO | CAPTURE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | SIP | | | | | | | |
| 0x0C | ISC_PFE_CFG0 | 7:0 | CONT | MODE[2:0] | | | FPOL | PPOL | VPOL | HPOL |
| | | 15:8 | | | ROWEN | COLEN | CCIR10_8N | CCIR_CRC | CCIR656 | GATED |
| | | 23:16 | SKIPCNT[7:0] | | | | | | | |
| | | 31:24 | REP | BPS[2:0] | | | CCIR_REP | | | |
| 0x10 | ISC_PFE_CFG1 | 7:0 | COLMIN[7:0] | | | | | | | |
| | | 15:8 | COLMIN[15:8] | | | | | | | |
| | | 23:16 | COLMAX[7:0] | | | | | | | |
| | | 31:24 | COLMAX[15:8] | | | | | | | |
| 0x14 | ISC_PFE_CFG2 | 7:0 | ROWMIN[7:0] | | | | | | | |
| | | 15:8 | ROWMIN[15:8] | | | | | | | |
| | | 23:16 | ROWMAX[7:0] | | | | | | | |
| | | 31:24 | ROWMAX[15:8] | | | | | | | |
| 0x18 | ISC_CLKEN | 7:0 | | | | | | | MCEN | ICEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | ISC_CLKDIS | 7:0 | | | | | | | MCDIS | ICDIS |
| | | 15:8 | | | | | | | MCSWRST | ICSWRST |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | ISC_CLKSR | 7:0 | | | | | | | MCSR | ICSR |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | SIP | | | | | | | |
| 0x24 | ISC_CLKCFG | 7:0 | ICDIV[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | ICSEL |
| | | 23:16 | MCDIV[7:0] | | | | | | | |
| | | 31:24 | | | | | | | MCSEL[1:0] | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x28 | ISC_INTEN | 7:0 | | | DIS | SWRST | | | HD | VD |
| | | 15:8 | | | HISCLR | HISDONE | | | LDONE | DDONE |
| | | 23:16 | | | | RERR | | | | WERR |
| | | 31:24 | | | | CCIRERR | HDTO | VDTO | DAOV | VFPOV |
| 0x2C | ISC_INTDIS | 7:0 | | | DIS | SWRST | | | HD | VD |
| | | 15:8 | | | HISCLR | HISDONE | | | LDONE | DDONE |
| | | 23:16 | | | | RERR | | | | WERR |
| | | 31:24 | | | | CCIRERR | HDTO | VDTO | DAOV | VFPOV |
| 0x30 | ISC_INTMASK | 7:0 | | | DIS | SWRST | | | HD | VD |
| | | 15:8 | | | HISCLR | HISDONE | | | LDONE | DDONE |
| | | 23:16 | | | | RERR | | | | WERR |
| | | 31:24 | | | | CCIRERR | HDTO | VDTO | DAOV | VFPOV |
| 0x34 | ISC_INTSR | 7:0 | | | DIS | SWRST | | | HD | VD |
| | | 15:8 | | | HISCLR | HISDONE | | | LDONE | DDONE |
| | | 23:16 | | | | RERR | | WERRID[1:0] | | WERR |
| | | 31:24 | | | | CCIRERR | HDTO | VDTO | DAOV | VFPOV |
| 0x38 ... 0x57 | Reserved | | | | | | | | | |
| 0x58 | ISC_WB_CTRL | 7:0 | | | | | | | | ENABLE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x5C | ISC_WB_CFG | 7:0 | | | | | | | BAYCFG[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x60 | ISC_WB_O_RGR | 7:0 | ROFST[7:0] | | | | | | | |
| | | 15:8 | | | | ROFST[12:8] | | | | |
| | | 23:16 | GROFST[7:0] | | | | | | | |
| | | 31:24 | | | | GROFST[12:8] | | | | |
| 0x64 | ISC_WB_O_BGB | 7:0 | BOFST[7:0] | | | | | | | |
| | | 15:8 | | | | BOFST[12:8] | | | | |
| | | 23:16 | GBOFST[7:0] | | | | | | | |
| | | 31:24 | | | | GBOFST[12:8] | | | | |
| 0x68 | ISC_WB_G_RGR | 7:0 | RGAIN[7:0] | | | | | | | |
| | | 15:8 | | | | RGAIN[12:8] | | | | |
| | | 23:16 | GRGAIN[7:0] | | | | | | | |
| | | 31:24 | | | | GRGAIN[12:8] | | | | |
| 0x6C | ISC_WB_G_BGB | 7:0 | BGAIN[7:0] | | | | | | | |
| | | 15:8 | | | | BGAIN[12:8] | | | | |
| | | 23:16 | GBGAIN[7:0] | | | | | | | |
| | | 31:24 | | | | GBGAIN[12:8] | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x70 | ISC_CFA_CTRL | 7:0 | | | | | | | | ENABLE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x74 | ISC_CFA_CFG | 7:0 | | | | EITPOL | | | BAYCFG[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x78 | ISC_CC_CTRL | 7:0 | | | | | | | | ENABLE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x7C | ISC_CC_RR_RG | 7:0 | RRGAIN[7:0] | | | | | | | |
| | | 15:8 | | | | | RRGAIN[11:8] | | | |
| | | 23:16 | RGGAIN[7:0] | | | | | | | |
| | | 31:24 | | | | | RGGAIN[11:8] | | | |
| 0x80 | ISC_CC_RB_OR | 7:0 | RBGAIN[7:0] | | | | | | | |
| | | 15:8 | | | | | RBGAIN[11:8] | | | |
| | | 23:16 | ROFST[7:0] | | | | | | | |
| | | 31:24 | | | | ROFST[12:8] | | | | |
| 0x84 | ISC_CC_GR_GG | 7:0 | GRGAIN[7:0] | | | | | | | |
| | | 15:8 | | | | | GRGAIN[11:8] | | | |
| | | 23:16 | GGGAIN[7:0] | | | | | | | |
| | | 31:24 | | | | | GGGAIN[11:8] | | | |
| 0x88 | ISC_CC_GB_OG | 7:0 | GBGAIN[7:0] | | | | | | | |
| | | 15:8 | | | | | GBGAIN[11:8] | | | |
| | | 23:16 | ROFST[7:0] | | | | | | | |
| | | 31:24 | | | | ROFST[12:8] | | | | |
| 0x8C | ISC_CC_BR_BG | 7:0 | BRGAIN[7:0] | | | | | | | |
| | | 15:8 | | | | | BRGAIN[11:8] | | | |
| | | 23:16 | BGGAIN[7:0] | | | | | | | |
| | | 31:24 | | | | | BGGAIN[11:8] | | | |
| 0x90 | ISC_CC_BB_OB | 7:0 | BBGAIN[7:0] | | | | | | | |
| | | 15:8 | | | | | BBGAIN[11:8] | | | |
| | | 23:16 | BOFST[7:0] | | | | | | | |
| | | 31:24 | | | | BOFST[12:8] | | | | |
| 0x94 | ISC_GAM_CTRL | 7:0 | | | | BIPART | RENABLE | GENABLE | BENABLE | ENABLE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x98 | ISC_GAM_BENTRY 0 | 7:0 | BSLOPE[7:0] | | | | | | | |
| | | 15:8 | | | | | | | BSLOPE[9:8] | |
| | | 23:16 | BCONSTANT[7:0] | | | | | | | |
| | | 31:24 | | | | | | | BCONSTANT[9:8] | |
| ... | | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0194 | ISC_GAM_BENTRY63 | 7:0 | BSLOPE[7:0] | | | | | | | | |
| | | 15:8 | | | | | | | BSLOPE[9:8] | | |
| | | 23:16 | BCONSTANT[7:0] | | | | | | | | |
| | | 31:24 | | | | | | | BCONSTANT[9:8] | | |
| 0x0198 | ISC_GAM_GENTRY0 | 7:0 | GSLOPE[7:0] | | | | | | | | |
| | | 15:8 | | | | | | | GSLOPE[9:8] | | |
| | | 23:16 | GCONSTANT[7:0] | | | | | | | | |
| | | 31:24 | | | | | | | GCONSTANT[9:8] | | |
| ... | | | | | | | | | | | |
| 0x0294 | ISC_GAM_GENTRY63 | 7:0 | GSLOPE[7:0] | | | | | | | | |
| | | 15:8 | | | | | | | GSLOPE[9:8] | | |
| | | 23:16 | GCONSTANT[7:0] | | | | | | | | |
| | | 31:24 | | | | | | | GCONSTANT[9:8] | | |
| 0x0298 | ISC_GAM_RENTRY0 | 7:0 | RSLOPE[7:0] | | | | | | | | |
| | | 15:8 | | | | | | | RSLOPE[9:8] | | |
| | | 23:16 | RCONSTANT[7:0] | | | | | | | | |
| | | 31:24 | | | | | | | RCONSTANT[9:8] | | |
| ... | | | | | | | | | | | |
| 0x0394 | ISC_GAM_RENTRY63 | 7:0 | RSLOPE[7:0] | | | | | | | | |
| | | 15:8 | | | | | | | RSLOPE[9:8] | | |
| | | 23:16 | RCONSTANT[7:0] | | | | | | | | |
| | | 31:24 | | | | | | | RCONSTANT[9:8] | | |
| 0x0398 | ISC_CSC_CTRL | 7:0 | | | | | | | | | ENABLE |
| | | 15:8 | | | | | | | | | |
| | | 23:16 | | | | | | | | | |
| | | 31:24 | | | | | | | | | |
| 0x039C | ISC_CSC_YR_YG | 7:0 | YRGAIN[7:0] | | | | | | | | |
| | | 15:8 | | | | | YRGAIN[11:8] | | | | |
| | | 23:16 | YGGAIN[7:0] | | | | | | | | |
| | | 31:24 | | | | | YGGAIN[11:8] | | | | |
| 0x03A0 | ISC_CSC_YB_OY | 7:0 | YBGAIN[7:0] | | | | | | | | |
| | | 15:8 | | | | | YBGAIN[11:8] | | | | |
| | | 23:16 | YOFST[7:0] | | | | | | | | |
| | | 31:24 | | | | | | YOFST[10:8] | | | |
| 0x03A4 | ISC_CSC_CBR_CBG | 7:0 | CBRGAIN[7:0] | | | | | | | | |
| | | 15:8 | | | | | CBRGAIN[11:8] | | | | |
| | | 23:16 | CBGGAIN[7:0] | | | | | | | | |
| | | 31:24 | | | | | CBGGAIN[11:8] | | | | |
| 0x03A8 | ISC_CSC_CBB_OCB | 7:0 | CBBGAIN[7:0] | | | | | | | | |
| | | 15:8 | | | | | CBBGAIN[11:8] | | | | |
| | | 23:16 | CBOFST[7:0] | | | | | | | | |
| | | 31:24 | | | | | | CBOFST[10:8] | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x03AC | ISC_CSC_CRR_CRG | 7:0 | CRRGAIN[7:0] | | | | | | | |
| | | 15:8 | | | | | CRRGAIN[11:8] | | | |
| | | 23:16 | CRGGAIN[7:0] | | | | | | | |
| | | 31:24 | | | | | CRGGAIN[11:8] | | | |
| 0x03B0 | ISC_CSC_CRB_OCR | 7:0 | CRBGAIN[7:0] | | | | | | | |
| | | 15:8 | | | | | CRBGAIN[11:8] | | | |
| | | 23:16 | CROFST[7:0] | | | | | | | |
| | | 31:24 | | | | | | CROFST[10:8] | | |
| 0x03B4 | ISC_CBC_CTRL | 7:0 | | | | | | | | ENABLE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03B8 | ISC_CBC_CFG | 7:0 | | | | | | CCIRMODE[1:0] | | CCIR |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03BC | ISC_CBC_BRIGHT | 7:0 | BRIGHT[7:0] | | | | | | | |
| | | 15:8 | | | | | | BRIGHT[10:8] | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03C0 | ISC_CBC_CONTRAST | 7:0 | CONTRAST[7:0] | | | | | | | |
| | | 15:8 | | | | | CONTRAST[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03C4 | ISC_SUB422_CTRL | 7:0 | | | | | | | | ENABLE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03C8 | ISC_SUB422_CFG | 7:0 | | | FILTER[1:0] | | | CCIRMODE[1:0] | | CCIR |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03CC | ISC_SUB420_CTRL | 7:0 | | | | FILTER | | | | ENABLE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03D0 | ISC_RLP_CFG | 7:0 | | | | | MODE[3:0] | | | |
| | | 15:8 | ALPHA[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03D4 | ISC_HIS_CTRL | 7:0 | | | | | | | | ENABLE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x03D8 | ISC_HIS_CFG | 7:0 | | | BAYSEL[1:0] | | | MODE[2:0] | | |
| | | 15:8 | | | | | | | | RAR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03DC ... 0x03DF | Reserved | | | | | | | | | |
| 0x03E0 | ISC_DCFG | 7:0 | | | YMBSIZE[1:0] | | | IMODE[2:0] | | |
| | | 15:8 | | | | | | | CMBSIZE[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03E4 | ISC_DCTRL | 7:0 | DONE | FIELD | WB | IE | | DVIEW[1:0] | | DE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03E8 | ISC_DNDA | 7:0 | NDA[5:0] | | | | | | | |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x03EC | ISC_DAD0 | 7:0 | AD0[7:0] | | | | | | | |
| | | 15:8 | AD0[15:8] | | | | | | | |
| | | 23:16 | AD0[23:16] | | | | | | | |
| | | 31:24 | AD0[31:24] | | | | | | | |
| 0x03F0 | ISC_DST0 | 7:0 | ST0[7:0] | | | | | | | |
| | | 15:8 | ST0[15:8] | | | | | | | |
| | | 23:16 | ST0[23:16] | | | | | | | |
| | | 31:24 | ST0[31:24] | | | | | | | |
| 0x03F4 | ISC_DAD1 | 7:0 | AD1[7:0] | | | | | | | |
| | | 15:8 | AD1[15:8] | | | | | | | |
| | | 23:16 | AD1[23:16] | | | | | | | |
| | | 31:24 | AD1[31:24] | | | | | | | |
| 0x03F8 | ISC_DST1 | 7:0 | ST1[7:0] | | | | | | | |
| | | 15:8 | ST1[15:8] | | | | | | | |
| | | 23:16 | ST1[23:16] | | | | | | | |
| | | 31:24 | ST1[31:24] | | | | | | | |
| 0x03FC | ISC_DAD2 | 7:0 | AD2[7:0] | | | | | | | |
| | | 15:8 | AD2[15:8] | | | | | | | |
| | | 23:16 | AD2[23:16] | | | | | | | |
| | | 31:24 | AD2[31:24] | | | | | | | |
| 0x0400 | ISC_DST2 | 7:0 | ST2[7:0] | | | | | | | |
| | | 15:8 | ST2[15:8] | | | | | | | |
| | | 23:16 | ST2[23:16] | | | | | | | |
| | | 31:24 | ST2[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0404 ... 0x040F | Reserved | | | | | | | | | |
| 0x0410 | ISC_HIS_ENTRY0 | 7:0 | COUNT[7:0] | | | | | | | |
| | | 15:8 | COUNT[15:8] | | | | | | | |
| | | 23:16 | | | | | COUNT[19:16] | | | |
| | | 31:24 | | | | | | | | |
| ... | | | | | | | | | | |
| 0x0C0C | ISC_HIS_ENTRY511 | 7:0 | COUNT[7:0] | | | | | | | |
| | | 15:8 | COUNT[15:8] | | | | | | | |
| | | 23:16 | | | | | COUNT[19:16] | | | |
| | | 31:24 | | | | | | | | |

### 51.6.1 ISC Control Enable Register 0

**Name:** ISC_CTRLEN
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Enables the corresponding command.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | FUPPRO | |
| Access | | | | | | | W | |
| Reset | | | | | | | – | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | HISCLR | HISREQ | UPPRO | CAPTURE |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

**Bit 9 – FUPPRO** Force Update Color Profile

**Bit 3 – HISCLR** Histogram Clear

**Bit 2 – HISREQ** Histogram Request

**Bit 1 – UPPRO** Update Profile

**Bit 0 – CAPTURE** Capture Input Stream Command

### 51.6.2 ISC Control Disable Register 0

**Name:** ISC_CTRLDIS
**Offset:** 0x04
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Performs the corresponding command.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SWRST |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DISABLE |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 8 – SWRST** Software Reset

**Bit 0 – DISABLE** Capture Disable

### 51.6.3 ISC Control Status Register 0

**Name:** ISC_CTRLSR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SIP | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | FIELD | | HISREQ | UPPRO | CAPTURE |
| Access | | | | R | | R | R | R |
| Reset | | | | 0 | | 0 | 0 | 0 |

**Bit 31 – SIP** Synchronization In Progress

| Value | Description |
|---|---|
| 0 | The double domain synchronization is terminated. |
| 1 | The double domain synchronization is in progress. |

**Bit 4 – FIELD** Field Status (only relevant when the video stream is interlaced)

| Value | Description |
|---|---|
| 0 | The current field/segment is a top field |
| 1 | The current field/segment is a bottom field. |

**Bit 2 – HISREQ** Histogram Request Pending

| Value | Description |
|---|---|
| 0 | There is no histogram pending request. |
| 1 | Indicates that the histogram request is still pending. |

**Bit 1 – UPPRO** Profile Update Pending

| Value | Description |
|---|---|
| 0 | There is no profile update pending request. |
| 1 | Indicates that the profile update request is still pending. |

**Bit 0 – CAPTURE** Capture pending

| Value | Description |
|---|---|
| 0 | Capture mode is disabled. |
| 1 | Capture is pending. |

### 51.6.4 ISC Parallel Front End Configuration 0 Register

**Name:** ISC_PFE_CFG0
**Offset:** 0x0C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | REP | BPS[2:0] | | | CCIR_REP | | | |
| Access | R/W | R/W | R/W | R/W | R/W | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SKIPCNT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | ROWEN | COLEN | CCIR10_8N | CCIR_CRC | CCIR656 | GATED |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CONT | MODE[2:0] | | | FPOL | PPOL | VPOL | HPOL |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – REP** Up Multiply with Replication

| Value | Description |
|---|---|
| 0 | Unused bits are stuck at 0. |
| 1 | Unused bits are copied from MSB. |

**Bits 30:28 – BPS[2:0]** Bits Per Sample

| Value | Name | Description |
|---|---|---|
| 0 | TWELVE | 12-bit input |
| 1 | ELEVEN | 11-bit input |
| 2 | TEN | 10-bit input |
| 3 | NINE | 9-bit input |
| 4 | EIGHT | 8-bit input |

**Bit 27 – CCIR_REP** CCIR Replication

| Value | Description |
|---|---|
| 0 | Unused bits are stuck at 0. |
| 1 | Unused bits are copied from MSB. |

**Bits 23:16 – SKIPCNT[7:0]** Frame Skipping Counter

**Bit 13 – ROWEN** Row Cropping Enable

| Value | Description |
|---|---|
| 0 | Row Cropping is disabled. |
| 1 | Row Cropping is enabled. |

**Bit 12 – COLEN** Column Cropping Enable

| Value | Description |
|---|---|
| 0 | Column Cropping is disabled. |
| 1 | Column Cropping is enabled. |

**Bit 11 – CCIR10_8N** CCIR 10 bits or 8 bits

| Value | Description |
|---|---|
| 0 | 8-bit mode. |
| 1 | 10-bit mode. |

**Bit 10 – CCIR_CRC** CCIR656 CRC Decoder

| Value | Description |
|---|---|
| 0 | Embedded CRC is discarded. |
| 1 | Embedded CRC is decoded. |

**Bit 9 – CCIR656** CCIR656 input mode

| Value | Description |
|---|---|
| 0 | HSYNC and VSYNC signals are used to synchronize the input stream. |
| 1 | Embedded synchronization is used. |

**Bit 8 – GATED** Gated input clock

| Value | Description |
|---|---|
| 0 | The external pixel clock is free running. |
| 1 | The external pixel clock is gated. |

**Bit 7 – CONT** Continuous Acquisition

| Value | Description |
|---|---|
| 0 | Single Shot mode. |
| 1 | Video mode. |

**Bits 6:4 – MODE[2:0]** Parallel Front End Mode

| Value | Name | Description |
|---|---|---|
| 0 | PROGRESSIVE | Video source is progressive. |
| 1 | DF_TOP | Video source is interlaced, two fields are captured starting with top field. |
| 2 | DF_BOTTOM | Video source is interlaced, two fields are captured starting with bottom field. |
| 3 | DF_IMMEDIATE | Video source is interlaced, two fields are captured immediately. |
| 4 | SF_TOP | Video source is interlaced, one field is captured starting with the top field. |

| Value | Name | Description |
|---|---|---|
| 5 | SF_BOTTOM | Video source is interlaced, one field is captured starting with the bottom field. |
| 6 | SF_IMMEDIATE | Video source is interlaced, one field is captured starting immediately. |

**Bit 3 – FPOL** Field Polarity

| Value | Description |
|---|---|
| 0 | Top field is sampled when F value is 0; Bottom field is sampled when F value is 1. |
| 1 | Top field is sampled when F value is 1; Bottom field is sampled when F value is 0. |

**Bit 2 – PPOL** Pixel Clock Polarity

| Value | Description |
|---|---|
| 0 | The pixel stream is sampled on the rising edge of the pixel clock. |
| 1 | The pixel stream is sampled on the falling edge of the pixel clock. |

**Bit 1 – VPOL** Vertical Synchronization Polarity

| Value | Description |
|---|---|
| 0 | VSYNC signal is active high, i.e. valid pixels are sampled when VSYNC is asserted. |
| 1 | VSYNC signal is active low, i.e. valid pixels are sampled when VSYNC is deasserted. |

**Bit 0 – HPOL** Horizontal Synchronization Polarity

| Value | Description |
|---|---|
| 0 | HSYNC signal is active high, i.e. valid pixels are sampled when HSYNC is asserted. |
| 1 | HSYNC signal is active low, i.e. valid pixels are sampled when HSYNC is deasserted. |

### 51.6.5 ISC Parallel Front End Configuration 1 Register

**Name:** ISC_PFE_CFG1
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | COLMAX[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | COLMAX[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COLMIN[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COLMIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – COLMAX[15:0]** Column Maximum Limit
Horizontal ending position of the cropping area.

**Bits 15:0 – COLMIN[15:0]** Column Minimum Limit
Horizontal starting position of the cropping area.

### 51.6.6 ISC Parallel Front End Configuration 2 Register

**Name:** ISC_PFE_CFG2
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ROWMAX[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ROWMAX[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ROWMIN[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ROWMIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – ROWMAX[15:0]** Row Maximum Limit
Vertical ending position of the cropping area.

**Bits 15:0 – ROWMIN[15:0]** Row Minimum Limit
Vertical starting position of the cropping area.

### 51.6.7 ISC Clock Enable Register

**Name:** ISC_CLKEN
**Offset:** 0x18
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | MCEN | ICEN |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

**Bit 1 – MCEN** Master Clock Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the master clock. |

**Bit 0 – ICEN** ISP Clock Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the ISP clock. |

### 51.6.8 ISC Clock Disable Register

**Name:** ISC_CLKDIS
**Offset:** 0x1C
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Performs the corresponding command.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | MCSWRST | ICSWRST |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | MCDIS | ICDIS |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

**Bit 9 – MCSWRST** Master Clock Software Reset

**Bit 8 – ICSWRST** ISP Clock Software Reset

**Bit 1 – MCDIS** Master Clock Disable

**Bit 0 – ICDIS** ISP Clock Disable

### 51.6.9 ISC Clock Status Register

**Name:** ISC_CLKSR
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SIP | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | MCSR | ICSR |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

**Bit 31 – SIP** Synchronization In Progress

| Value | Description |
|---|---|
| 0 | The double domain synchronization operation is over. |
| 1 | The double domain synchronization operation is in progress. |

**Bit 1 – MCSR** Master Clock Status Register

| Value | Description |
|---|---|
| 0 | The master clock is disabled. |
| 1 | The master clock is enabled. |

**Bit 0 – ICSR** ISP Clock Status Register

| Value | Description |
|---|---|
| 0 | The ISP clock is disabled. |
| 1 | The ISP clock is enabled. |

### 51.6.10 ISC Clock Configuration Register

**Name:** ISC_CLKCFG
**Offset:** 0x24
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | MCSEL[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MCDIV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ICSEL |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ICDIV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 25:24 – MCSEL[1:0]** Master Clock Reference Clock Selection

| Value | Description |
|---|---|
| 0 | HCLOCK is selected. |
| 1 | ISCCLK is selected. |
| 2 | GCK is selected. |

**Bits 23:16 – MCDIV[7:0]** Master Clock Divider

$$f_{\mathrm{mc}} = \frac{f_{\mathrm{mcref}}}{\mathrm{MCDIV} + 1}$$

**Bit 8 – ICSEL** ISP Clock Selection

| Value | Description |
|---|---|
| 0 | HCLOCK is selected. |
| 1 | ISCCLK is selected. |

**Bits 7:0 – ICDIV[7:0]** ISP Clock Divider

$$f_{\mathrm{cc}} = \frac{f_{\mathrm{ccref}}}{\mathrm{ICDIV} + 1}$$

### 51.6.11 ISC Interrupt Enable Register

**Name:** ISC_INTEN
**Offset:** 0x28
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Enables the interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | CCIRERR | HDTO | VDTO | DAOV | VFPOV |
| Access | | | | W | W | W | W | W |
| Reset | | | | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | RERR | | | | WERR |
| Access | | | | W | | | | W |
| Reset | | | | – | | | | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | HISCLR | HISDONE | | | LDONE | DDONE |
| Access | | | W | W | | | W | W |
| Reset | | | – | – | | | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DIS | SWRST | | | HD | VD |
| Access | | | W | W | | | W | W |
| Reset | | | – | – | | | – | – |

**Bit 28 – CCIRERR** CCIR Decoder Error Interrupt Enable

**Bit 27 – HDTO** Horizontal Synchronization Timeout Interrupt Enable

**Bit 26 – VDTO** Vertical Synchronization Timeout Interrupt Enable

**Bit 25 – DAOV** Data Overflow Interrupt Enable

**Bit 24 – VFPOV** Vertical Front Porch Overflow Interrupt Enable

**Bit 20 – RERR** Read Channel Error Interrupt Enable

**Bit 16 – WERR** Write Channel Error Interrupt Enable

**Bit 13 – HISCLR** Histogram Clear Interrupt Enable

**Bit 12 – HISDONE** Histogram Completed Interrupt Enable

**Bit 9 – LDONE** DMA List Done Interrupt Enable

**Bit 8 – DDONE** DMA Done Interrupt Enable

**Bit 5 – DIS** Disable Completed Interrupt Enable

**Bit 4 – SWRST** Software Reset Completed Interrupt Enable

**Bit 1 – HD** Horizontal Synchronization Detection Interrupt Enable

**Bit 0 – VD** Vertical Synchronization Detection Interrupt Enable

### 51.6.12 ISC Interrupt Disable Register

**Name:** ISC_INTDIS
**Offset:** 0x2C
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:
0: No effect.

1: Disables the interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | CCIRERR | HDTO | VDTO | DAOV | VFPOV |
| Access | | | | W | W | W | W | W |
| Reset | | | | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | RERR | | | | WERR |
| Access | | | | W | | | | W |
| Reset | | | | – | | | | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | HISCLR | HISDONE | | | LDONE | DDONE |
| Access | | | W | W | | | W | W |
| Reset | | | – | – | | | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DIS | SWRST | | | HD | VD |
| Access | | | W | W | | | W | W |
| Reset | | | – | – | | | – | – |

**Bit 28 – CCIRERR** CCIR Decoder Error Interrupt Disable

**Bit 27 – HDTO** Horizontal Synchronization Timeout Interrupt Disable

**Bit 26 – VDTO** Vertical Synchronization Timeout Interrupt Disable

**Bit 25 – DAOV** Data Overflow Interrupt Disable

**Bit 24 – VFPOV** Vertical Front Porch Overflow Interrupt Disable

**Bit 20 – RERR** Read Channel Error Interrupt Disable

**Bit 16 – WERR** Write Channel Error Interrupt Disable

**Bit 13 – HISCLR** Histogram Clear Interrupt Disable

**Bit 12 – HISDONE** Histogram Completed Interrupt Disable

**Bit 9 – LDONE** DMA List Done Interrupt Disable

**Bit 8 – DDONE** DMA Done Interrupt Disable

**Bit 5 – DIS** Disable Completed Interrupt Disable

**Bit 4 – SWRST** Software Reset Completed Interrupt Disable

**Bit 1 – HD** Horizontal Synchronization Detection Interrupt Disable

**Bit 0 – VD** Vertical Synchronization Detection Interrupt Disable

### 51.6.13 ISC Interrupt Mask Register

**Name:** ISC_INTMASK
**Offset:** 0x30
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:
0: The interrupt is disabled.

1: The interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | CCIRERR | HDTO | VDTO | DAOV | VFPOV |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | RERR | | | | WERR |
| Access | | | | R | | | | R |
| Reset | | | | 0 | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | HISCLR | HISDONE | | | LDONE | DDONE |
| Access | | | R | R | | | R | R |
| Reset | | | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DIS | SWRST | | | HD | VD |
| Access | | | R | R | | | R | R |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bit 28 – CCIRERR** CCIR Decoder Error Interrupt Mask

**Bit 27 – HDTO** Horizontal Synchronization Timeout Interrupt Mask

**Bit 26 – VDTO** Vertical Synchronization Timeout Interrupt Mask

**Bit 25 – DAOV** Data Overflow Interrupt Mask

**Bit 24 – VFPOV** Vertical Front Porch Overflow Interrupt Mask

**Bit 20 – RERR** Read Channel Error Interrupt Mask

**Bit 16 – WERR** Write Channel Error Interrupt Mask

**Bit 13 – HISCLR** Histogram Clear Interrupt Mask

**Bit 12 – HISDONE** Histogram Completed Interrupt Mask

**Bit 9 – LDONE** DMA List Done Interrupt Mask

**Bit 8 – DDONE** DMA Done Interrupt Mask

**Bit 5 – DIS** Disable Completed Interrupt Mask

**Bit 4 – SWRST** Software Reset Completed Interrupt Mask

**Bit 1 – HD** Horizontal Synchronization Detection Interrupt Mask

**Bit 0 – VD** Vertical Synchronization Detection Interrupt Mask

### 51.6.14 ISC Interrupt Status Register

**Name:** ISC_INTSR
**Offset:** 0x34
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | CCIRERR | HDTO | VDTO | DAOV | VFPOV |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | RERR | | WERRID[1:0] | | WERR |
| Access | | | | R | | R | R | R |
| Reset | | | | 0 | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | HISCLR | HISDONE | | | LDONE | DDONE |
| Access | | | R | R | | | R | R |
| Reset | | | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DIS | SWRST | | | HD | VD |
| Access | | | R | R | | | R | R |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bit 28 – CCIRERR** CCIR Decoder Error Interrupt (cleared on read)

| Value | Description |
|---|---|
| 0 | No CCIR CRC error detected since the last read of the Interrupt Status register. |
| 1 | A CCIR CRC error has been detected. |

**Bit 27 – HDTO** Horizontal Synchronization Timeout Interrupt (cleared on read)

| Value | Description |
|---|---|
| 0 | A horizontal synchronization is detected. |
| 1 | No horizontal synchronization is detected. |

**Bit 26 – VDTO** Vertical Synchronization Timeout Interrupt (cleared on read)

| Value | Description |
|---|---|
| 0 | A vertical synchronization is detected. |
| 1 | No vertical synchronization is detected. |

**Bit 25 – DAOV** Data Overflow Interrupt (cleared on read)

| Value | Description |
|---|---|
| 0 | No data overflow error occurred since the last reset of the Interrupt Status register. |
| 1 | A data overflow occurred. |

**Bit 24 – VFPOV** Vertical Front Porch Overflow Interrupt (cleared on read)

| Value | Description |
|---|---|
| 0 | No vertical front porch error occurred since the last read of the Interrupt Status register. |
| 1 | The vertical synchronization has been detected but the DMA channel is still busy. |

**Bit 20 – RERR** Read Channel Error Interrupt (cleared on read)

| Value | Description |
|---|---|
| 0 | No read channel error since the last read of the Interrupt Status register. |
| 1 | A read channel error occurred when the ISC read the descriptor. |

**Bits 18:17 – WERRID[1:0]** Write Channel Error Identifier (cleared on read)

| Value | Name | Description |
|---|---|---|
| 0 | CH0 | An error occurred for Channel 0 (RAW/RGB/Y) |
| 1 | CH1 | An error occurred for Channel 1 (CbCr/Cb) |
| 2 | CH2 | An error occurred for Channel 2 (Cr) |
| 3 | WB | Write back channel error |

**Bit 16 – WERR** Write Channel Error Interrupt (cleared on read)

| Value | Description |
|---|---|
| 0 | No write channel error since the last read of the Interrupt Status register. |
| 1 | A write channel error occurred. |

**Bit 13 – HISCLR** Histogram Clear Interrupt (cleared on read)

| Value | Description |
|---|---|
| 0 | No Histogram Clear Interrupt has been raised since the last read of the Interrupt Status register. |
| 1 | The Histogram Clear Interrupt has occurred. |

**Bit 12 – HISDONE** Histogram Completed Interrupt (cleared on read)

| Value | Description |
|---|---|
| 0 | No Histogram Completed Interrupt has been raised since the last read of the Interrupt Status register. |
| 1 | The Histogram Completed Interrupt has occurred. |

**Bit 9 – LDONE** DMA List Done Interrupt (cleared on read)

| Value | Description |
|---|---|
| 0 | No DMA List Done Interrupt has occurred since the last read of the Interrupt Status register. |
| 1 | The DMA List Done Interrupt has occurred. |

**Bit 8 – DDONE** DMA Done Interrupt (cleared on read)

| Value | Description |
|---|---|
| 0 | No DMA Transfer Done Interrupt has occurred since the last read of the Interrupt Status register. |
| 1 | The DMA Transfer Done Interrupt has occurred. |

**Bit 5 – DIS** Disable Completed Interrupt (cleared on read)

| Value | Description |
|---|---|
| 0 | The disable has not occurred since the last read of the Interrupt Status register. |
| 1 | The disable has completed. |

**Bit 4 – SWRST** Software Reset Completed Interrupt (cleared on read)

| Value | Description |
|---|---|
| 0 | No software reset completion since the last read of the Interrupt Status register. |
| 1 | The software reset has completed. |

**Bit 1 – HD** Horizontal Synchronization Detected Interrupt (cleared on read)

| Value | Description |
|---|---|
| 0 | No horizontal synchronization detection since the last read of the Interrupt Status register. |
| 1 | A horizontal synchronization has been detected. |

**Bit 0 – VD** Vertical Synchronization Detected Interrupt (cleared on read)

| Value | Description |
|---|---|
| 0 | No vertical synchronization detection since the last read of the Interrupt Status register. |
| 1 | A vertical synchronization has been detected. |

### 51.6.15 ISC White Balance Control Register

**Name:** ISC_WB_CTRL
**Offset:** 0x58
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ENABLE |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – ENABLE** White Balance Enable

| Value | Description |
|---|---|
| 0 | The white balance is disabled. |
| 1 | The white balance is enabled. |

### 51.6.16 ISC White Balance Configuration Register

**Name:** ISC_WB_CFG
**Offset:** 0x5C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | BAYCFG[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bits 1:0 – BAYCFG[1:0]** White Balance Bayer Configuration (Pixel Color Pattern)

| Value | Name | Description |
|---|---|---|
| 0 | GRGR | Starting Row configuration is G R G R (Red Row). |
| 1 | RGRG | Starting Row configuration is R G R G (Red Row). |
| 2 | GBGB | Starting Row configuration is G B G B (Blue Row). |
| 3 | BGBG | Starting Row configuration is B G B G (Blue Row). |

### 51.6.17 ISC White Balance Offset for R, GR Register

**Name:** ISC_WB_O_RGR
**Offset:** 0x60
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | GROFST[12:8] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GROFST[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | ROFST[12:8] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ROFST[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 28:16 – GROFST[12:0]** Offset Green Component for Red Row (signed 13 bits 1:12:0)

**Bits 12:0 – ROFST[12:0]** Offset Red Component (signed 13 bits 1:12:0)

### 51.6.18 ISC White Balance Offset for B and GB Register

**Name:** ISC_WB_O_BGB
**Offset:** 0x64
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | GBOFST[12:8] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GBOFST[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | BOFST[12:8] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BOFST[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 28:16 – GBOFST[12:0]** Offset Green Component for Blue Row (signed 13 bits, 1:12:0)

**Bits 12:0 – BOFST[12:0]** Offset Blue Component (signed 13 bits, 1:12:0)

### 51.6.19 ISC White Balance Gain for R, GR Register

**Name:** ISC_WB_G_RGR
**Offset:** 0x68
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | GRGAIN[12:8] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GRGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | RGAIN[12:8] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 28:16 – GRGAIN[12:0]** Green Component (Red row) Gain (unsigned 13 bits, 0:4:9)

**Bits 12:0 – RGAIN[12:0]** Red Component Gain (unsigned 13 bits, 0:4:9)

### 51.6.20 ISC White Balance Gain for B, GB Register

**Name:** ISC_WB_G_BGB
**Offset:** 0x6C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | GBGAIN[12:8] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GBGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | BGAIN[12:8] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 28:16 – GBGAIN[12:0]** Green Component (Blue row) Gain (unsigned 13 bits, 0:4:9)

**Bits 12:0 – BGAIN[12:0]** Blue Component Gain (unsigned 13 bits, 0:4:9)

### 51.6.21 ISC Color Filter Array Control Register

**Name:** ISC_CFA_CTRL
**Offset:** 0x70
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ENABLE |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – ENABLE** Color Filter Array Interpolation Enable

| Value | Description |
|---|---|
| 0 | Color Filter Array Interpolation is disabled. |
| 1 | Color Filter Array Interpolation is enabled. |

### 51.6.22 ISC Color Filter Array Configuration Register

**Name:** ISC_CFA_CFG
**Offset:** 0x74
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | EITPOL | | | BAYCFG[1:0] | |
| Access | | | | R/W | | | R/W | R/W |
| Reset | | | | 0 | | | 0 | 0 |

**Bit 4 – EITPOL** Edge Interpolation

| Value | Description |
|---|---|
| 0 | Edges are not interpolated. |
| 1 | Edge interpolation is performed. |

**Bits 1:0 – BAYCFG[1:0]** Color Filter Array Pattern

| Value | Name | Description |
|---|---|---|
| 0 | GRGR | Starting row configuration is G R G R (red row). |
| 1 | RGRG | Starting row configuration is R G R G (red row). |
| 2 | GBGB | Starting row configuration is G B G B (blue row). |
| 3 | BGBG | Starting row configuration is B G B G (blue row). |

### 51.6.23 ISC Color Correction Control Register

**Name:** ISC_CC_CTRL
**Offset:** 0x78
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ENABLE |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – ENABLE** Color Correction Enable

| Value | Description |
|---|---|
| 0 | Color correction is disabled. |
| 1 | Color correction is enabled. |

### 51.6.24 ISC Color Correction RR RG Register

**Name:** ISC_CC_RR_RG
**Offset:** 0x7C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RGGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RGGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RRGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RRGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27:16 – RGGAIN[11:0]** Green Gain for Red Component (signed 12 bits, 1:3:8)

**Bits 11:0 – RRGAIN[11:0]** Red Gain for Red Component (signed 12 bits, 1:3:8)

### 51.6.25 ISC Color Correction RB OR Register

**Name:** ISC_CC_RB_OR
**Offset:** 0x80
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | ROFST[12:8] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ROFST[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RBGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RBGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 28:16 – ROFST[12:0]** Red Component Offset (signed 13 bits, 1:12:0)

**Bits 11:0 – RBGAIN[11:0]** Blue Gain for Red Component (signed 12 bits, 1:3:8)

### 51.6.26 ISC Color Correction GR GG Register

**Name:** ISC_CC_GR_GG
**Offset:** 0x84
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | GGGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GGGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | GRGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | GRGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27:16 – GGGAIN[11:0]** Green Gain for Green Component (signed 12 bits, 1:3:8)

**Bits 11:0 – GRGAIN[11:0]** Red Gain for Green Component (signed 12 bits, 1:3:8)

### 51.6.27 ISC Color Correction GB OG Register

**Name:** ISC_CC_GB_OG
**Offset:** 0x88
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | ROFST[12:8] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ROFST[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | GBGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | GBGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 28:16 – ROFST[12:0]** Green Component Offset (signed 13 bits, 1:12:0)

**Bits 11:0 – GBGAIN[11:0]** Blue Gain for Green Component (signed 12 bits, 1:3:8)

### 51.6.28 ISC Color Correction BR BG Register

**Name:** ISC_CC_BR_BG
**Offset:** 0x8C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | BGGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BGGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | BRGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BRGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27:16 – BGGAIN[11:0]** Green Gain for Blue Component (signed 12 bits, 1:3:8)

**Bits 11:0 – BRGAIN[11:0]** Red Gain for Blue Component (signed 12 bits, 1:3:8)

### 51.6.29 ISC Color Correction BB OB Register

**Name:** ISC_CC_BB_OB
**Offset:** 0x90
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | BOFST[12:8] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BOFST[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | BBGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BBGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 28:16 – BOFST[12:0]** Blue Component Offset (signed 13 bits, 1:12:0)

**Bits 11:0 – BBGAIN[11:0]** Blue Gain for Blue Component (signed 12 bits, 1:3:8)

### 51.6.30 ISC Gamma Correction Control Register

**Name:** ISC_GAM_CTRL
**Offset:** 0x94
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | BIPART | RENABLE | GENABLE | BENABLE | ENABLE |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bit 4 – BIPART** Bipartite Table Configuration

| Value | Description |
|---|---|
| 0 | Bipartite table is disabled. There are 64 points of interpolation from 0 to 4095. |
| 1 | Bipartite table is enabled. There are 32 points of interpolation (spacing is 8) from 0 to 255, then there are 30 points of interpolation from 256 to 4095. |

**Bit 3 – RENABLE** Gamma Correction Enable for R Channel

| Value | Description |
|---|---|
| 0 | 12-bit to 10-bit compression is performed skipping two bits. |
| 1 | Piecewise interpolation is used to perform 12-bit to 10-bit compression for the red channel. |

**Bit 2 – GENABLE** Gamma Correction Enable for G Channel

| Value | Description |
|---|---|
| 0 | 12-bit to 10-bit compression is performed skipping two bits. |
| 1 | Piecewise interpolation is used to perform 12-bit to 10-bit compression for the green channel. |

**Bit 1 – BENABLE** Gamma Correction Enable for B Channel

| Value | Description |
|---|---|
| 0 | 12-bit to 10-bit compression is performed skipping two bits. |
| 1 | Piecewise interpolation is used to perform 12-bit to 10-bit compression for the blue channel. |

**Bit 0 – ENABLE** Gamma Correction Enable

| Value | Description |
|---|---|
| 0 | Gamma correction is disabled. |
| 1 | Gamma correction is enabled. |

### 51.6.31 ISC Gamma Correction Blue Entry Register x [x=0..63]

**Name:** ISC_GAM_BENTRYx
**Offset:** 0x98 + x*0x04 [x=0..63]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | BCONSTANT[9:8] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BCONSTANT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | BSLOPE[9:8] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BSLOPE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 25:16 – BCONSTANT[9:0]** Blue Color Constant for Piecewise Interpolation (unsigned 10 bits 0:10:0)

**Bits 9:0 – BSLOPE[9:0]** Blue Color Slope for Piecewise Interpolation (signed 10 bits 1:3:6)

### 51.6.32 ISC Gamma Correction Green Entry Register x [x=0..63]

**Name:** ISC_GAM_GENTRYx
**Offset:** 0x0198 + x*0x04 [x=0..63]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | GCONSTANT[9:8] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GCONSTANT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | GSLOPE[9:8] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | GSLOPE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 25:16 – GCONSTANT[9:0]** Green Color Constant for Piecewise Interpolation (unsigned 10 bits 0:10:0)

**Bits 9:0 – GSLOPE[9:0]** Green Color Slope for Piecewise Interpolation (signed 10 bits 1:3:6)

### 51.6.33 ISC Gamma Correction Red Entry Register x [x=0..63]

**Name:** ISC_GAM_RENTRYx
**Offset:** 0x0298 + x*0x04 [x=0..63]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RCONSTANT[9:8] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RCONSTANT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RSLOPE[9:8] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RSLOPE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 25:16 – RCONSTANT[9:0]** Red Color Constant for Piecewise Interpolation (unsigned 10 bits 0:10:0)

**Bits 9:0 – RSLOPE[9:0]** Red Color Slope for Piecewise Interpolation (signed 10 bits 1:3:6)

### 51.6.34 ISC Color Space Conversion Control Register

**Name:** ISC_CSC_CTRL
**Offset:** 0x398
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ENABLE |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – ENABLE** RGB to YCbCr Color Space Conversion Enable

| Value | Description |
|---|---|
| 0 | Color space conversion is disabled. |
| 1 | Color space conversion is enabled. |

### 51.6.35 ISC Color Space Conversion YR YG Register

**Name:** ISC_CSC_YR_YG
**Offset:** 0x39C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | YGGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YGGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | YRGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | YRGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27:16 – YGGAIN[11:0]** Green Gain for Luminance (signed 12 bits 1:3:8)

**Bits 11:0 – YRGAIN[11:0]** Reg Gain for Luminance (signed 12 bits 1:3:8)

### 51.6.36 ISC Color Space Conversion YB OY Register

**Name:** ISC_CSC_YB_OY
**Offset:** 0x3A0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | YOFST[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YOFST[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | YBGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | YBGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 26:16 – YOFST[10:0]** Luminance Offset (11 bits signed 1:10:0)

**Bits 11:0 – YBGAIN[11:0]** Blue Gain for Luminance Component (12 bits signed 1:3:8)

### 51.6.37 ISC Color Space Conversion CBR CBG Register

**Name:** ISC_CSC_CBR_CBG
**Offset:** 0x3A4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CBGGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CBGGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CBRGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CBRGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27:16 – CBGGAIN[11:0]** Green Gain for Blue Chrominance (signed 12 bits 1:3:8)

**Bits 11:0 – CBRGAIN[11:0]** Red Gain for Blue Chrominance (signed 12 bits, 1:3:8)

### 51.6.38 ISC Color Space Conversion CBB OCB Register

**Name:** ISC_CSC_CBB_OCB
**Offset:** 0x3A8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CBOFST[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CBOFST[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CBBGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CBBGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 26:16 – CBOFST[10:0]** Blue Chrominance Offset (signed 11 bits 1:10:0)

**Bits 11:0 – CBBGAIN[11:0]** Blue Gain for Blue Chrominance (signed 12 bits 1:3:8)

### 51.6.39 ISC Color Space Conversion CRR CRG Register

**Name:** ISC_CSC_CRR_CRG
**Offset:** 0x3AC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CRGGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CRGGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CRRGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CRRGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27:16 – CRGGAIN[11:0]** Green Gain for Red Chrominance (signed 12 bits 1:3:8)

**Bits 11:0 – CRRGAIN[11:0]** Red Gain for Red Chrominance (signed 12 bits 1:3:8)

### 51.6.40 ISC Color Space Conversion CRB OCR Register

**Name:** ISC_CSC_CRB_OCR
**Offset:** 0x3B0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CROFST[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CROFST[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CRBGAIN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CRBGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 26:16 – CROFST[10:0]** Red Chrominance Offset (signed 11 bits 1:10:0)

**Bits 11:0 – CRBGAIN[11:0]** Blue Gain for Red Chrominance (signed 12 bits 1:3:8)

### 51.6.41 ISC Contrast And Brightness Control Register

**Name:** ISC_CBC_CTRL
**Offset:** 0x3B4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ENABLE |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – ENABLE** Contrast and Brightness Control Enable

| Value | Description |
|---|---|
| 0 | Contrast and brightness control are disabled. |
| 1 | Contrast and brightness control are enabled. |

### 51.6.42 ISC Contrast And Brightness Configuration Register

**Name:** ISC_CBC_CFG
**Offset:** 0x3B8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CCIRMODE[1:0] | | CCIR |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 2:1 – CCIRMODE[1:0]** CCIR656 Byte Ordering

| Value | Name | Description |
|---|---|---|
| 0 | CBY | Byte ordering Cb0, Y0, Cr0, Y1 |
| 1 | CRY | Byte ordering Cr0, Y0, Cb0, Y1 |
| 2 | YCB | Byte ordering Y0, Cb0, Y1, Cr0 |
| 3 | YCR | Byte ordering Y0, Cr0, Y1, Cb0 |

**Bit 0 – CCIR** CCIR656 Stream Enable

| Value | Description |
|---|---|
| 0 | Raw mode. |
| 1 | CCIR656 stream. |

### 51.6.43 ISC Contrast And Brightness, Brightness Register

**Name:** ISC_CBC_BRIGHT
**Offset:** 0x3BC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | BRIGHT[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BRIGHT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 10:0 – BRIGHT[10:0]** Image Brightness Control (signed 11 bits 1:10:0)
Brightness value is added or subtracted from the luminance Y data.

### 51.6.44 ISC Contrast And Brightness, Contrast Register

**Name:** ISC_CBC_CONTRAST
**Offset:** 0x3C0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CONTRAST[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CONTRAST[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 11:0 – CONTRAST[11:0]** Contrast (unsigned 12 bits 0:4:8)
Adjusts the image contrast by multiplying with YCbCr data.

### 51.6.45 ISC Subsampling 4:4:4 to 4:2:2 Control Register

**Name:** ISC_SUB422_CTRL
**Offset:** 0x3C4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ENABLE |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – ENABLE** 4:4:4 to 4:2:2 Chrominance Horizontal Subsampling Filter Enable

| Value | Description |
|---|---|
| 0 | Subsampler is disabled. |
| 1 | Subsampler is enabled. |

### 51.6.46 ISC Subsampling 4:4:4 to 4:2:2 Configuration Register

**Name:** ISC_SUB422_CFG
**Offset:** 0x3C8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | FILTER[1:0] | | | CCIRMODE[1:0] | | CCIR |
| Access | | | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | 0 | 0 | | 0 | 0 | 0 |

**Bits 5:4 – FILTER[1:0]** Low Pass Filter Selection

| Value | Name | Description |
|---|---|---|
| 0 | FILT0CO | Cosited, {1} |
| 1 | FILT1CE | Centered {1, 1} |
| 2 | FILT2CO | Cosited {1,2,1} |
| 3 | FILT3CE | Centered {1, 3, 3, 1} |

**Bits 2:1 – CCIRMODE[1:0]** CCIR656 Byte Ordering

| Value | Name | Description |
|---|---|---|
| 0 | CBY | Byte ordering Cb0, Y0, Cr0, Y1 |
| 1 | CRY | Byte ordering Cr0, Y0, Cb0, Y1 |
| 2 | YCB | Byte ordering Y0, Cb0, Y1, Cr0 |
| 3 | YCR | Byte ordering Y0, Cr0, Y1, Cb0 |

**Bit 0 – CCIR** CCIR656 Input Stream

| Value | Description |
|---|---|
| 0 | Raw mode. |
| 1 | CCIR mode. |

### 51.6.47 ISC Subsampling 4:2:2 to 4:2:0 Control Register

**Name:** ISC_SUB420_CTRL
**Offset:** 0x3CC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | FILTER | | | | ENABLE |
| Access | | | | R/W | | | | R/W |
| Reset | | | | 0 | | | | 0 |

**Bit 4 – FILTER** Interlaced or Progressive Chrominance Filter

| Value | Description |
|---|---|
| 0 | Progressive filter {0.5, 0.5}. |
| 1 | Field-dependent filter, top field filter is {0.75, 0.25}, bottom field filter is {0.25, 0.75}. |

**Bit 0 – ENABLE** 4:2:2 to 4:2:0 Vertical Subsampling Filter Enable (Center Aligned)

| Value | Description |
|---|---|
| 0 | Subsampler disabled. |
| 1 | Subsampler enabled. |

### 51.6.48 ISC Rounding, Limiting and Packing Configuration Register

**Name:** ISC_RLP_CFG
**Offset:** 0x3D0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ALPHA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MODE[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 15:8 – ALPHA[7:0]** Alpha Value for Alpha-enabled RGB Mode

**Bits 3:0 – MODE[3:0]** Rounding, Limiting and Packing Mode

| Value | Name | Description |
|---|---|---|
| 0 | DAT8 | 8-bit data |
| 1 | DAT9 | 9-bit data |
| 2 | DAT10 | 10-bit data |
| 3 | DAT11 | 11-bit data |
| 4 | DAT12 | 12-bit data |
| 5 | DATY8 | 8-bit luminance only |
| 6 | DATY10 | 10-bit luminance only |
| 7 | ARGB444 | 12-bit RGB+4-bit Alpha (MSB) |
| 8 | ARGB555 | 15-bit RGB+1-bit Alpha (MSB) |
| 9 | RGB565 | 16-bit RGB |
| 10 | ARGB32 | 24-bits RGB mode+8-bit Alpha |
| 11 | YYCC | YCbCr mode (full range, [0–255]) |
| 12 | YYCC_LIMITED | YCbCr mode (limited range) |

### 51.6.49 ISC Histogram Control Register

**Name:** ISC_HIS_CTRL
**Offset:** 0x3D4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ENABLE |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – ENABLE** Histogram Sub Module Enable

| Value | Description |
|---|---|
| 0 | Histogram disabled. |
| 1 | Histogram enabled. |

### 51.6.50 ISC Histogram Configuration Register

**Name:** ISC_HIS_CFG
**Offset:** 0x3D8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | RAR |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | BAYSEL[1:0] | | | MODE[2:0] | | |
| Access | | | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | 0 | 0 | | 0 | 0 | 0 |

**Bit 8 – RAR** Histogram Reset After Read

| Value | Description |
|---|---|
| 0 | Reset after read mode is disabled. |
| 1 | Reset after read mode is enabled. |

**Bits 5:4 – BAYSEL[1:0]** Bayer Color Component Selection

| Value | Name | Description |
|---|---|---|
| 0 | GRGR | Starting row configuration is G R G R (red row) |
| 1 | RGRG | Starting row configuration is R G R G (red row) |
| 2 | GBGB | Starting row configuration is G B G B (blue row) |
| 3 | BGBG | Starting row configuration is B G B G (blue row) |

**Bits 2:0 – MODE[2:0]** Histogram Operating Mode

| Value | Name | Description |
|---|---|---|
| 0 | GR | Gr sampling |
| 1 | R | R sampling |
| 2 | GB | Gb sampling |
| 3 | B | B sampling |
| 4 | Y | Luminance-only mode |

| Value | Name | Description |
|---|---|---|
| 5 | RAW | Raw sampling |
| 6 | YCCIR656 | Luminance only with CCIR656 10-bit or 8-bit mode |

### 51.6.51 ISC DMA Configuration Register

**Name:** ISC_DCFG
**Offset:** 0x3E0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CMBSIZE[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | YMBSIZE[1:0] | | | IMODE[2:0] | | |
| Access | | | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | 0 | 0 | | 0 | 0 | 0 |

**Bits 9:8 – CMBSIZE[1:0]** DMA Memory Burst Size C channel

| Value | Name | Description |
|---|---|---|
| 0 | SINGLE | DMA single access |
| 1 | BEATS4 | 4-beat burst access |
| 2 | BEATS8 | 8-beat burst access |
| 3 | BEATS16 | 16-beat burst access |

**Bits 5:4 – YMBSIZE[1:0]** DMA Memory Burst Size Y channel

| Value | Name | Description |
|---|---|---|
| 0 | SINGLE | DMA single access |
| 1 | BEATS4 | 4-beat burst access |
| 2 | BEATS8 | 8-beat burst access |
| 3 | BEATS16 | 16-beat burst access |

**Bits 2:0 – IMODE[2:0]** DMA Input Mode Selection

| Value | Name | Description |
|---|---|---|
| 0 | PACKED8 | 8 bits, single channel packed |
| 1 | PACKED16 | 16 bits, single channel packed |
| 2 | PACKED32 | 32 bits, single channel packed |

| Value | Name | Description |
|---|---|---|
| 3 | YC422SP | 32 bits, dual channel |
| 4 | YC422P | 32 bits, triple channel |
| 5 | YC420SP | 32 bits, dual channel |
| 6 | YC420P | 32 bits, triple channel |

### 51.6.52 ISC DMA Control Register

**Name:** ISC_DCTRL
**Offset:** 0x3E4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DONE | FIELD | WB | IE | | DVIEW[1:0] | | DE |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bit 7 – DONE** Descriptor Processing Status
This bit appears only in the descriptor located in memory and can be used only if WB (Write Back) is set.

| Value | Description |
|---|---|
| 0 | Descriptor not processed yet. |
| 1 | Descriptor processed. |

**Bit 6 – FIELD** Value of Captured Frame Field Signal
This bit is only relevant for interlaced content. It appears only in the descriptor located in memory and can be used only if WB (Write Back) is set.

| Value | Description |
|---|---|
| 0 | Field value is 0. |
| 1 | Field value is 1. |

**Bit 5 – WB** Write Back Operation Enable

| Value | Description |
|---|---|
| 0 | Write Back operation is skipped. |
| 1 | Write Back operation is performed. |

**Bit 4 – IE** Interrupt Enable

| Value | Description |
|---|---|
| 0 | DMA Done interrupt is generated. |
| 1 | DMA Done interrupt is not set. |

**Bits 2:1 – DVIEW[1:0]** Descriptor View

| Value | Name | Description |
|---|---|---|
| 0 | PACKED | Address {0} Stride {0} are updated |
| 1 | SEMIPLANAR | Address {0,1} Stride {0,1} are updated |
| 2 | PLANAR | Address {0,1,2} Stride {0,1,2} are updated |

**Bit 0 – DE** Descriptor Enable

| Value | Description |
|---|---|
| 0 | Descriptor disabled. |
| 1 | Descriptor enabled. |

### 51.6.53 ISC DMA Descriptor Address Register

**Name:** ISC_DNDA
**Offset:** 0x3E8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NDA[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NDA[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NDA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NDA[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – NDA[29:0]** Next Descriptor Address Register

### 51.6.54 ISC DMA Address 0 Register

**Name:** ISC_DAD0
**Offset:** 0x3EC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | AD0[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | AD0[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | AD0[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | AD0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – AD0[31:0]** Channel 0 Address

### 51.6.55 ISC DMA Stride 0 Register

**Name:** ISC_DST0
**Offset:** 0x3F0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ST0[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ST0[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ST0[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ST0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ST0[31:0]** Channel 0 Stride

### 51.6.56 ISC DMA Address 1 Register

**Name:** ISC_DAD1
**Offset:** 0x3F4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | AD1[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | AD1[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | AD1[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | AD1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – AD1[31:0]** Channel 1 Address

### 51.6.57 ISC DMA Stride 1 Register

**Name:** ISC_DST1
**Offset:** 0x3F8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ST1[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ST1[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ST1[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ST1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ST1[31:0]** Channel 1 Stride

### 51.6.58 ISC DMA Address 2 Register

**Name:** ISC_DAD2
**Offset:** 0x3FC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | AD2[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | AD2[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | AD2[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | AD2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – AD2[31:0]** Channel 2 Address

### 51.6.59 ISC DMA Stride 2 Register

**Name:** ISC_DST2
**Offset:** 0x400
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ST2[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ST2[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ST2[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ST2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ST2[31:0]** Channel 2 Stride

### 51.6.60 ISC Histogram Entry x [x=0..511]

**Name:** ISC_HIS_ENTRYx
**Offset:** 0x0410 + x*0x04 [x=0..511]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | COUNT[19:16] | | | |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 19:0 – COUNT[19:0]** Entry Counter

# 52. Controller Area Network (MCAN)

## 52.1 Description

The Controller Area Network (MCAN) performs communication according to CAN Protocol Version 2.0 Part A,B and Bosch CAN FD Specification V1.0 (Non-ISO). Additional transceiver hardware is required for connection to the physical layer.

All functions concerning the handling of messages are implemented by the Rx Handler and the Tx Handler. The Rx Handler manages message acceptance filtering, the transfer of received messages from the CAN core to the Message RAM, as well as providing receive message status information. The Tx Handler is responsible for the transfer of transmit messages from the Message RAM to the CAN core, as well as providing transmit status information.

Acceptance filtering is implemented by a combination of up to 128 filter elements, where each element can be configured as a range, as a bit mask, or as a dedicated ID filter.

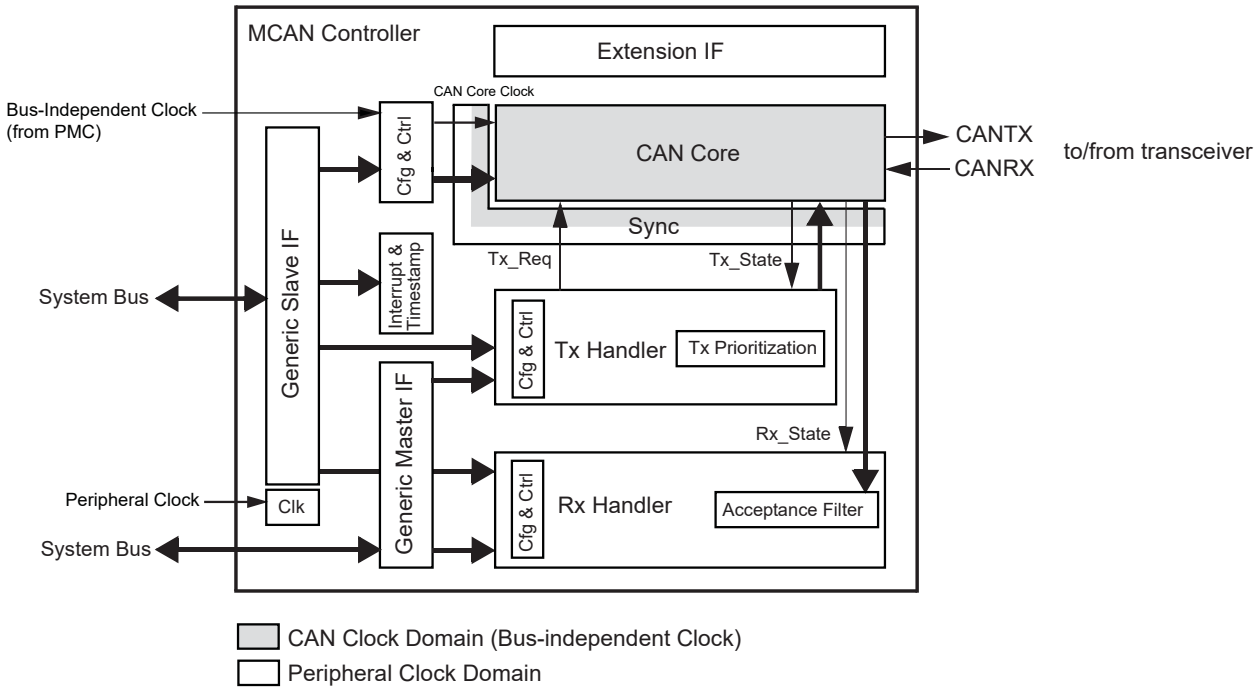
**WARNING** MCAN implements the non-ISO CAN FD frame format and therefore does not pass the CAN FD Conformance Test according to ISO 16845-1:2016.

## 52.2 Embedded Characteristics

- Classic CAN (2.0 A,B) in Conformance with ISO11898-1:2015 and ISO16845-1:2016
- CAN FD Frame Format according to Bosch CAN FD Specification V1.0 (Non-ISO)
- CAN-FD with up to 64 Data Bytes Supported
- CAN Error Logging
- AUTOSAR Optimized
- SAE J1939 Optimized
- Improved Acceptance Filtering
- Two Configurable Receive FIFOs
- Separate Signalling on Reception of High Priority Messages
- Up to 64 Dedicated Receive Buffers
- Up to 32 Dedicated Transmit Buffers
- Configurable Transmit FIFO
- Configurable Transmit Queue
- Configurable Transmit Event FIFO
- Direct Message RAM Access for Processor
- Multiple MCANs May Share the Same Message RAM
- Programmable Loop-back Test Mode
- Maskable Module Interrupts
- Support for Asynchronous CAN and System Bus Clocks
- Power-down Support
- Debug on CAN Support

## 52.3 Block Diagram

**Figure 52-1. MCAN Block Diagram**




**Note:** Refer to section “Power Management Controller (PMC)” for details about the bus-independent clock (GCLK).

## 52.4 Product Dependencies

### 52.4.1 I/O Lines

The pins used to interface to the compliant external devices can be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the CAN pins to their peripheral functions.

### 52.4.2 Power Management

The MCAN can be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the MCAN clock.

In order to achieve a stable function of the MCAN, the system bus clock must always be faster than or equal to the CAN clock.

It is recommended to use the CAN clock at frequencies of 20, 40 or 80 MHz. To achieve these frequencies, PMC GCLK must select the UPLLCK (480 MHz) as source clock and divide by 24,12, or 6. GCLK allows the system bus and processor clock to be modified without affecting the bit rate communication.

### 52.4.3 Interrupt Sources

The two MCAN interrupt lines (MCAN_INT0, MCAN_INT1) are connected on internal sources of the Interrupt Controller.

Using the MCAN interrupts requires the Interrupt Controller to be programmed first.

Interrupt sources can be routed either to MCAN_INT0 or to MCAN_INT1. By default, all interrupt sources are routed to interrupt line MCAN_INT0/1. By programming MCAN_ILE.EINT0 and MCAN_ILE.EINT1, the interrupt sources can be enabled or disabled separately.

#### 52.4.4 Address Configuration

The LSBs [bits 15:2] for each section of the CAN Message RAM are configured in the respective buffer configuration registers as detailed in Message RAM.

The MSBs [bits 31:16] of the CAN Message RAM for CAN0 and CAN1 are configured in 0x00200000.

## 52.5 Functional Description

### 52.5.1 Operating Modes

#### 52.5.1.1 Software Initialization

Software initialization is started by setting bit MCAN_CCCR.INIT, either by software or by a hardware reset, when an uncorrected bit error was detected in the Message RAM, or by going Bus_Off. While MCAN_CCCR.INIT is set, message transfer from and to the CAN bus is stopped and the status of the CAN bus output CANTX is recessive (HIGH). The counters of the Error Management Logic EML are unchanged. Setting MCAN_CCCR.INIT does not change any configuration register. Resetting MCAN_CCCR.INIT finishes the software initialization. Afterwards the Bit Stream Processor BSP synchronizes itself to the data transfer on the CAN bus by waiting for the occurrence of a sequence of 11 consecutive recessive bits (≡ Bus_Idle) before it can take part in bus activities and start the message transfer.

Access to the MCAN configuration registers is only enabled when both bits MCAN_CCCR.INIT and MCAN_CCCR.CCE are set (protected write).

MCAN_CCCR.CCE can only be configured when MCAN_CCCR.INIT = ‘1’. MCAN_CCCR.CCE is automatically cleared when MCAN_CCCR.INIT = ‘0’.

The following registers are cleared when MCAN_CCCR.CCE = ‘1’:

- High Priority Message Status (MCAN_HPMS)
- Receive FIFO 0 Status (MCAN_RXF0S)
- Receive FIFO 1 Status (MCAN_RXF1S)
- Transmit FIFO/Queue Status (MCAN_TXFQS)
- Transmit Buffer Request Pending (MCAN_TXBRP)
- Transmit Buffer Transmission Occurred (MCAN_TXBTO)
- Transmit Buffer Cancellation Finished (MCAN_TXBCF)
- Transmit Event FIFO Status (MCAN_TXEFS)

The Timeout Counter value MCAN_TOCV.TOC is loaded with the value configured by MCAN_TOCC.TOP when MCAN_CCCR.CCE = ‘1’.

In addition, the state machines of the Tx Handler and Rx Handler are held in idle state while MCAN_CCCR.CCE = ‘1’.

The following registers are only writeable while MCAN_CCCR.CCE = ‘0’

- Transmit Buffer Add Request (MCAN_TXBAR)
- Transmit Buffer Cancellation Request (MCAN_TXBCR)

MCAN_CCCR.TEST and MCAN_CCCR.MON can only be set when MCAN_CCCR.INIT = ‘1’ and MCAN_CCCR.CCE = ‘1’. Both bits may be cleared at any time. MCAN_CCCR.DAR can only be configured when MCAN_CCCR.INIT = ‘1’ and MCAN_CCCR.CCE = ‘1’.

#### 52.5.1.2 Normal Operation

Once the MCAN is initialized and MCAN_CCCR.INIT is cleared, the MCAN synchronizes itself to the CAN bus and is ready for communication.

After passing the acceptance filtering, received messages including Message ID and DLC are stored into a dedicated Rx Buffer or into Rx FIFO 0 or Rx FIFO 1.

For messages to be transmitted, dedicated Tx Buffers and/or a Tx FIFO or a Tx Queue can be initialized or updated. Automated transmission on reception of remote frames is not implemented.

#### 52.5.1.3 CAN FD Operation

There are two variants in the CAN FD frame format, first the CAN FD frame without bit rate switching where the data field of a CAN frame may be longer than 8 bytes. The second variant is the CAN FD frame where control field, data field, and CRC field of a CAN frame are transmitted with a higher bit rate than the beginning and the end of the frame.

The previously reserved bit in CAN frames with 11-bit identifiers and the first previously reserved bit in CAN frames with 29-bit identifiers will now be decoded as FDF bit. FDF = recessive signifies a CAN FD frame, FDF = dominant signifies a Classic CAN frame. In a CAN FD frame, the two bits following FDF, res and BRS, decide whether the bit rate inside of this CAN FD frame is switched. A CAN FD bit rate switch is signified by res = dominant and BRS = recessive. The coding of res = recessive is reserved for future expansion of the protocol. In case the MCAN receives a frame with FDF = recessive and res = recessive, it will signal a Protocol Exception Event by setting bit MCAN_PSR.PXE. When Protocol Exception Handling is enabled (MCAN_CCCR.PXHD = 0), this causes the operation state to change from Receiver (MCAN_PSR.ACT = 2) to Integrating (MCAN_PSR.ACT = 00) at the next sample point. In case Protocol Exception Handling is disabled (MCAN_CCCR.PXHD = 1), the MCAN will treat a recessive res bit as an form error and will respond with an error frame.

CAN FD operation is enabled by programming CCCR.FDOE. In case CCCR.FDOE = ‘1’, transmission and reception of CAN FD frames is enabled. Transmission and reception of Classic CAN frames is always possible. Whether a CAN FD frame or a Classic CAN frame is transmitted can be configured via bit FDF in the respective Tx Buffer element. With CCCR.FDOE = ‘0’, received frames are interpreted as Classic CAN frames, which leads to the transmission of an error frame when receiving a CAN FD frame. When CAN FD operation is disabled, no CAN FD frames are transmitted even if bit FDF of a Tx Buffer element is set. CCCR.FDOE and CCCR.BRSE can only be changed while CCCR.INIT and CCCR.CCE are both set.

With MCAN_CCCR.FDOE = 0, the setting of bits FDF and BRS is ignored and frames are transmitted in Classic CAN format. With MCAN_CCCR.FDOE = 1 and MCAN_CCCR.BRSE = 0, only bit FDF of a Tx Buffer element is evaluated. With MCAN_CCCR.FDOE = 1 and MCAN_CCCR.BRSE = 1, transmission of CAN FD frames with bit rate switching is enabled. All Tx Buffer elements with bits FDF and BRS set are transmitted in CAN FD format with bit rate switching.

A mode change during CAN operation is only recommended under the following conditions:

- The failure rate in the CAN FD data phase is significant higher than in the CAN FD arbitration phase. In this case disable the CAN FD bit rate switching option for transmissions.
- During system startup all nodes are transmitting according to ISO11898-1 until it is verified that they are able to communicate in CAN FD format. If this is true, all nodes switch to CAN FD operation.
- Wake-up messages in CAN Partial Networking have to be transmitted in Classic CAN format.

- End-of-line programming in case not all nodes are CAN FD-capable. Non-CAN FD nodes are held in Silent mode until programming has completed. Then all nodes revert to Classic CAN communication.

In the CAN FD format, the coding of the DLC differs from the standard CAN format. The DLC codes 0 to 8 have the same coding as in standard CAN, the codes 9 to 15, which in standard CAN all code a data field of 8 bytes, are coded according to the table below.

**Table 52-1. Coding of DLC in CAN FD**

| DLC | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| Number of Data Bytes | 12 | 16 | 20 | 24 | 32 | 48 | 64 |

In CAN FD frames, the bit timing will be switched inside the frame, after the BRS (Bit Rate Switch) bit, if this bit is recessive. Before the BRS bit, in the CAN FD arbitration phase, the nominal CAN bit timing is used as defined by the Nominal Bit Timing and Prescaler register (MCAN_NBTP). In the following CAN FD data phase, the data phase CAN bit timing is used as defined by the FastData Bit Timing and Prescaler register (MCAN_DBTP). The bit timing reverts back from the data phase timing at the CRC delimiter or when an error is detected, whichever occurs first.

The maximum configurable bit rate in the CAN FD data phase depends on the CAN core clock frequency. Example: with a CAN clock frequency of 20 MHz and the shortest configurable bit time of 4 $t_q$, the bit rate in the data phase is 5 Mbit/s.

In both data frame formats, CAN FD and CAN FD with bit rate switching, the value of the bit ESI (Error Status Indicator) is determined by the transmitter's error state at the start of the transmission. If the transmitter is error passive, ESI is transmitted recessive, else it is transmitted dominant.

#### 52.5.1.4 Transmitter Delay Compensation

During the data phase of a CAN FD transmission only one node is transmitting, all others are receivers. The length of the bus line has no impact. When transmitting via pin CANTX the protocol controller receives the transmitted data from its local CAN transceiver via pin CANRX. The received data is delayed by the transmitter delay. In case this delay is greater than TSEG1 (time segment before sample point), a bit error is detected. In order to enable a data phase bit time that is even shorter than the transmitter delay, the delay compensation is introduced. Without delay compensation, the bit rate in the data phase of a CAN FD frame is limited by the delay.

##### 52.5.1.4.1 Description

The MCAN protocol unit has implemented a delay compensation mechanism to compensate the delay, thereby enabling transmission with higher bit rates during the CAN FD data phase independent of the delay of a specific CAN transceiver.

To check for bit errors during the data phase, the delayed transmit data is compared against the received data at the secondary sample point. If a bit error is detected, the transmitter will react to this bit error at the next following regular sample point. During arbitration phase the delay compensation is always disabled.

The transmitter delay compensation enables configurations where the data bit time is shorter than the transmitter delay, it is described in detail in the new ISO11898-1. It is enabled by setting bit MCAN_DBTP.TDC.

The received bit is compared against the transmitted bit at the SSP. The SSP position is defined as the sum of the measured delay from the MCAN's transmit output CANTX through the transceiver to the receive input CANRX plus the transmitter delay compensation offset as configured by

MCAN_TDCR.TDCO. The transmitter delay compensation offset is used to adjust the position of the SSP inside the received bit (e.g. half of the bit time in the data phase). The position of the secondary sample point is rounded down to the next integer number of CAN core clock periods.

MCAN_PSR.TDCV shows the actual transmitter delay compensation value. MCAN_PSR.TDCV is cleared when MCAN_CCCR.INIT is set and is updated at each transmission of an FD frame while MCAN_DBTP.TDC is set.

The following boundary conditions have to be considered for the delay compensation implemented in the MCAN:

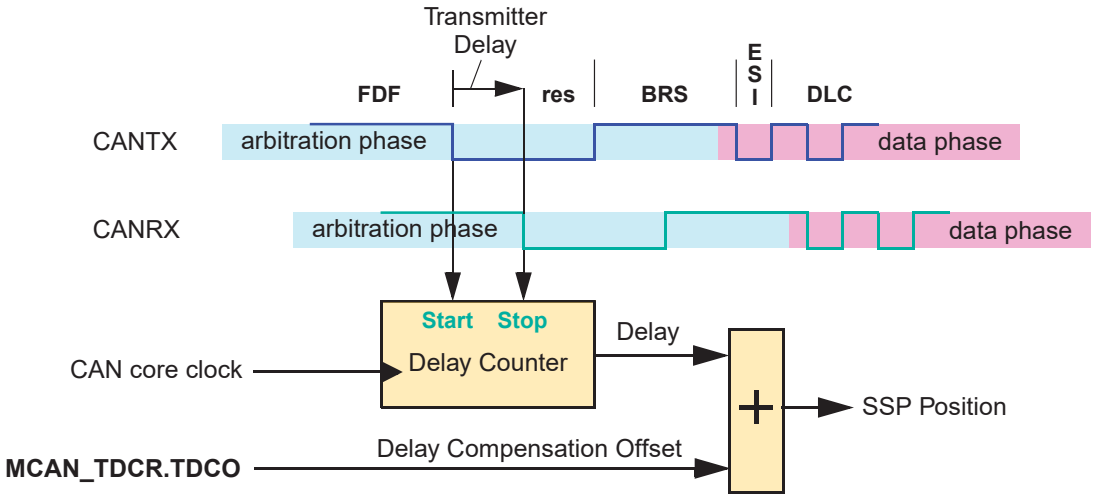
- The sum of the measured delay from CANTX to CANRX and the configured delay compensation offset MCAN_TDCR.TDCO has to be less than 6 bit times in the data phase.
- The sum of the measured delay from CANTX to CANRX and the configured delay compensation offset MCAN_TDCR.TDCO has to be less or equal 127 CAN core clock periods. In case this sum exceeds 127 CAN core clock periods, the maximum value of 127 CAN core clock periods is used for delay compensation.
- The data phase ends at the sample point of the CRC delimiter, that stops checking of receive bits at the SSPs.

#### 52.5.1.4.2 Transmitter Delay Measurement

If transmitter delay compensation is enabled by programming MCAN_DBTP.TDC = '1', the measurement is started within each transmitted CAN FD frame at the falling edge of bit FDF to bit res. The measurement is stopped when this edge is seen at the receive input CANRX of the transmitter.

The resolution of this measurement is one mtq.

**Figure 52-2. Transmitter Delay Measurement**




To avoid that a dominant glitch inside the received FDF bit ends the delay compensation measurement before the falling edge of the received res bit, resulting in a to early SSP position, the use of a transmitter delay compensation filter window can be enabled by programming MCAN_TDCR.TDCF.

This defines a minimum value for the SSP position. Dominant edges on CANRX, that would result in an earlier SSP position are ignored for transmitter delay measurement. The measurement is stopped when the SSP position is at least MCAN_TDCR.TDCF AND CANRX is low.

### 52.5.1.5 Restricted Operation Mode

In Restricted Operation mode, the node is able to receive data and remote frames and to give acknowledge to valid frames, but it does not send data frames, remote frames, active error frames, or

overload frames. In case of an error condition or overload condition, it does not send dominant bits, instead it waits for the occurrence of bus idle condition to resynchronize itself to the CAN communication. The error counters are not incremented. The processor can set the MCAN into Restricted Operation mode by setting bit MCAN_CCCR.ASM. The bit can only be set by the processor when both MCAN_CCCR.CCE and MCAN_CCCR.INIT are set to '1'. The bit can be reset by the processor at any time.

Restricted Operation mode is automatically entered when the Tx Handler was not able to read data from the Message RAM in time. To leave Restricted Operation mode, the processor has to reset MCAN_CCCR.ASM.

The Restricted Operation mode can be used in applications that adapt themselves to different CAN bit rates. In this case the application tests different bit rates and leaves the Restricted Operation mode after it has received a valid frame.

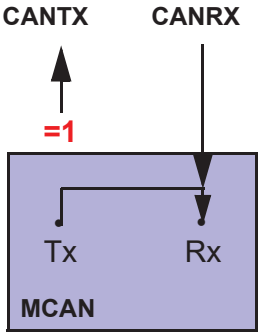
**Note:** The Restricted Operation Mode must not be combined with the Loop Back mode (internal or external).

#### 52.5.1.6 Bus Monitoring Mode

The MCAN is set in Bus Monitoring mode by setting MCAN_CCCR.MON. In Bus Monitoring mode (see ISO11898-1, 10.12 Bus monitoring), the MCAN is able to receive valid data frames and valid remote frames, but cannot start a transmission. In this mode, it sends only recessive bits on the CAN bus. If the MCAN is required to send a dominant bit (ACK bit, overload flag, active error flag), the bit is rerouted internally so that the MCAN monitors this dominant bit, although the CAN bus may remain in recessive state. In Bus Monitoring mode, the Tx Buffer Request Pending register (MCAN_TXBRP) is held in reset state.

The Bus Monitoring mode can be used to analyze the traffic on a CAN bus without affecting it by the transmission of dominant bits. The figure below shows the connection of signals CANTX and CANRX to the MCAN in Bus Monitoring mode.

**Figure 52-3. Pin Control in Bus Monitoring Mode**




**Bus Monitoring Mode**

#### 52.5.1.7 Disabled Automatic Retransmission

According to the CAN Specification (see ISO11898-1, 6.3.3 Recovery Management), the MCAN provides means for automatic retransmission of frames that have lost arbitration or that have been disturbed by errors during transmission. By default automatic retransmission is enabled. To support time-triggered communication as described in ISO 11898-1, chapter 9.2, the automatic retransmission may be disabled via MCAN_CCCR.DAR.

#### 52.5.1.7.1 Frame Transmission in DAR Mode

In DAR mode, all transmissions are automatically cancelled after they start on the CAN bus. A Tx Buffer's Tx Request Pending bit TXBRP.TRPx is reset after successful transmission, when a transmission has not yet been started at the point of cancellation, has been aborted due to lost arbitration, or when an error occurred during frame transmission.

- Successful transmission:
  Corresponding Tx Buffer Transmission Occurred bit MCAN_TXBTO.TOx set

  Corresponding Tx Buffer Cancellation Finished bit MCAN_TXBCF.CFx not set
- Successful transmission in spite of cancellation:
  Corresponding Tx Buffer Transmission Occurred bit MCAN_TXBTO.TOx set

  Corresponding Tx Buffer Cancellation Finished bit MCAN_TXBCF.CFx set
- Arbitration lost or frame transmission disturbed:
  Corresponding Tx Buffer Transmission Occurred bit MCAN_TXBTO.TOx not set

  Corresponding Tx Buffer Cancellation Finished bit MCAN_TXBCF.CFx set

In case of a successful frame transmission, and if storage of Tx events is enabled, a Tx Event FIFO element is written with Event Type ET = "10" (transmission in spite of cancellation).

### 52.5.1.8 Power-down (Sleep Mode)

The MCAN can be set into Power-down mode via bit MCAN_CCCR.CSR.

When all pending transmission requests have completed, the MCAN waits until bus idle state is detected. Then the MCAN sets MCAN_CCCR.INIT to prevent any further CAN transfers. Now the MCAN acknowledges that it is ready for power down by setting to one the bit MCAN_CCCR.CSA. In this state, before the clocks are switched off, further register accesses can be made. A write access to MCAN_CCCR.INIT will have no effect. Now the bus clock (peripheral clock) and the CAN core clock may be switched off.

To leave Power-down mode, the application has to turn on the MCAN clocks before clearing CC Control Register flag MCAN_CCCR.CSR. The MCAN will acknowledge this by clearing MCAN_CCCR.CSA. The application can then restart CAN communication by clearing the bit CCCR.INIT.

### 52.5.1.9 Test Modes

To enable write access to the MCAN Test register (MCAN_TEST) (see Section 7.6), bit MCAN_CCCR.TEST must be set. This allows the configuration of the test modes and test functions.

Four output functions are available for the CAN transmit pin CANTX by programming MCAN_TEST.TX. Additionally to its default function – the serial data output – it can drive the CAN Sample Point signal to monitor the MCAN's bit timing and it can drive constant dominant or recessive values. The actual value at pin CANRX can be read from MCAN_TEST.RX. Both functions can be used to check the CAN bus' physical layer.

Due to the synchronization mechanism between CAN clock and system bus clock domain, there may be a delay of several system bus clock periods between writing to MCAN_TEST.TX until the new configuration is visible at output pin CANTX. This applies also when reading input pin CANRX via MCAN_TEST.RX.

**Note:** Test modes should be used for production tests or self-test only. The software control for pin CANTX interferes with all CAN protocol functions. It is not recommended to use test modes for application.

#### 52.5.1.9.1 External Loop Back Mode

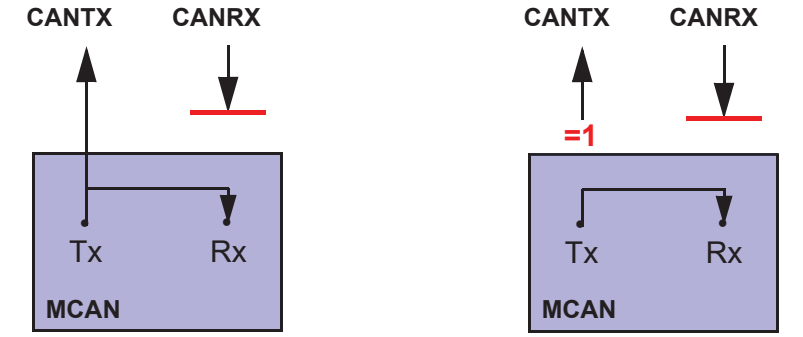
The MCAN can be set in External Loop Back mode by setting the bit MCAN_TEST.LBCK. In Loop Back mode, the MCAN treats its own transmitted messages as received messages and stores them (if they pass acceptance filtering) into an Rx Buffer or an Rx FIFO. The figure below shows the connection of signals CANTX and CANRX to the MCAN in External Loop Back mode.

This mode is provided for hardware self-test. To be independent from external stimulation, the MCAN ignores acknowledge errors (recessive bit sampled in the acknowledge slot of a data/remote frame) in Loop Back mode. In this mode, the MCAN performs an internal feedback from its Tx output to its Rx input. The actual value of the CANRX input pin is disregarded by the MCAN. The transmitted messages can be monitored at the CANTX pin.

#### 52.5.1.9.2 Internal Loop Back Mode

Internal Loop Back mode is entered by setting bits MCAN_TEST.LBCK and MCAN_CCCR.MON. This mode can be used for a “Hot Selftest”, meaning the MCAN can be tested without affecting a running CAN system connected to the pins CANTX and CANRX. In this mode, pin CANRX is disconnected from the MCAN, and pin CANTX is held recessive. The figure below shows the connection of CANTX and CANRX to the MCAN when Internal Loop Back mode is enabled.

**Figure 52-4. Pin Control in Loop Back Modes**




### 52.5.2 Timestamp Generation

For timestamp generation, the MCAN supplies a 16-bit wrap-around counter. A prescaler TSCC.TCP can be configured to clock the counter in multiples of CAN bit times (1…16). The counter is readable via MCAN_TSCV.TSC. A write access to the Timestamp Counter Value register (MCAN_TSCV) resets the counter to zero. When the timestamp counter wraps around, the interrupt flag MCAN_IR.TSW is set.

On start of frame reception / transmission, the counter value is captured and stored into the timestamp section of an Rx Buffer / Rx FIFO (RXTS[15:0]) or Tx Event FIFO (TXTS[15:0]) element.

By programming bit MCAN_TSCC.TSS, an external 16-bit timestamp can be used.

### 52.5.3 Timeout Counter

To signal timeout conditions for Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO, the MCAN supplies a 16-bit Timeout Counter. It operates as down-counter and uses the same prescaler controlled by TSCC.TCP as the Timestamp Counter. The Timeout Counter is configured via the Timeout Counter Configuration register (MCAN_TOCC). The actual counter value can be read from MCAN_TOCV.TOC. The Timeout Counter can only be started while MCAN_CCCR.INIT = ‘0’. It is stopped when MCAN_CCCR.INIT = ‘1’, e.g. when the MCAN enters Bus_Off state.

The operating mode is selected by MCAN_TOCC.TOS. When operating in Continuous mode, the counter starts when MCAN_CCCR.INIT is reset. A write to MCAN_TOCV presets the counter to the value configured by MCAN_TOCC.TOP and continues down-counting.

When the Timeout Counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by MCAN_TOCC.TOP. Down-counting is started when the first FIFO element is stored. Writing to MCAN_TOCV has no effect.

When the counter reaches zero, interrupt flag MCAN_IR.TOO is set. In Continuous mode, the counter is immediately restarted at MCAN_TOCC.TOP.

**Note:** The clock signal for the Timeout Counter is derived from the CAN Core's sample point signal. Therefore the point in time where the Timeout Counter is decremented may vary due to the synchronization / re-synchronization mechanism of the CAN Core. If the bit rate switch feature in CAN FD is used, the timeout counter is clocked differently in arbitration and data field.

### 52.5.4 Rx Handling

The Rx Handler controls the acceptance filtering, the transfer of received messages to the Rx Buffers or to one of the two Rx FIFOs, as well as the Rx FIFO's Put and Get Indices.

#### 52.5.4.1 Acceptance Filtering

The MCAN offers the possibility to configure two sets of acceptance filters, one for standard identifiers and one for extended identifiers. These filters can be assigned to an Rx Buffer or to Rx FIFO 0,1. For acceptance filtering each list of filters is executed from element #0 until the first matching element. Acceptance filtering stops at the first matching element. The following filter elements are not evaluated for this message.

The main features are:

- Each filter element can be configured as
  - range filter (from - to)
  - filter for one or two dedicated IDs
  - classic bit mask filter
- Each filter element is configurable for acceptance or rejection filtering
- Each filter element can be enabled / disabled individually
- Filters are checked sequentially, execution stops with the first matching filter element

Related configuration registers are:

- Global Filter Configuration (MCAN_GFC)
- Standard ID Filter Configuration (MCAN_SIDFC)
- Extended ID Filter Configuration (MCAN_XIDFC)
- Extended ID and Mask (MCAN_XIDAM)

Depending on the configuration of the filter element (SFEC/EFEC) a match triggers one of the following actions:

- Store received frame in FIFO 0 or FIFO 1
- Store received frame in Rx Buffer
- Store received frame in Rx Buffer and generate pulse at filter event pin
- Reject received frame
- Set High Priority Message interrupt flag (MCAN_IR.HPM)

- Set High Priority Message interrupt flag (MCAN_IR.HPM) and store received frame in FIFO 0 or FIFO 1

Acceptance filtering is started after the complete identifier has been received. After acceptance filtering has completed, and if a matching Rx Buffer or Rx FIFO has been found, the Message Handler starts writing the received message data in portions of 32 bit to the matching Rx Buffer or Rx FIFO. If the CAN protocol controller has detected an error condition (e.g. CRC error), this message is discarded with the following impact on the effected Rx Buffer or Rx FIFO:

- Rx Buffer
  New Data flag of matching Rx Buffer is not set, but Rx Buffer (partly) overwritten with received data. For error type, see MCAN_PSR.LEC and MCAN_PSR.DLEC.
- Rx FIFO
  Put index of matching Rx FIFO is not updated, but related Rx FIFO element (partly) overwritten with received data. For error type, see MCAN_PSR.LEC and MCAN_PSR.DLEC. In case the matching Rx FIFO is operated in Overwrite mode, the boundary conditions described in Rx FIFO Overwrite Mode have to be considered.
  **Note:** When an accepted message is written to one of the two Rx FIFOs, or into an Rx Buffer, the unmodified received identifier is stored independent of the filter(s) used. The result of the acceptance filter process is strongly depending on the sequence of configured filter elements.

#### 52.5.4.1.1 Range Filter

The filter matches for all received frames with Message IDs in the range defined by SF1ID/SF2ID resp. EF1ID/EF2ID.

There are two possibilities when range filtering is used together with extended frames:

- EFT = “00”: The Message ID of received frames is ANDed with MCAN_XIDAM before the range filter is applied.
- EFT = “11”: MCAN_XIDAM is not used for range filtering.

#### 52.5.4.1.2 Filter for Specific IDs

A filter element can be configured to filter for one or two specific Message IDs. To filter for one specific Message ID, the filter element has to be configured with SF1ID = SF2ID resp. EF1ID = EF2ID.

#### 52.5.4.1.3 Classic Bit Mask Filter

Classic bit mask filtering is intended to filter groups of Message IDs by masking single bits of a received Message ID. With classic bit mask filtering SF1ID/EF1ID is used as Message ID filter, while SF2ID/EF2ID is used as filter mask.

A zero bit at the filter mask will mask out the corresponding bit position of the configured ID filter, e.g. the value of the received Message ID at that bit position is not relevant for acceptance filtering. Only those bits of the received Message ID where the corresponding mask bits are one are relevant for acceptance filtering.

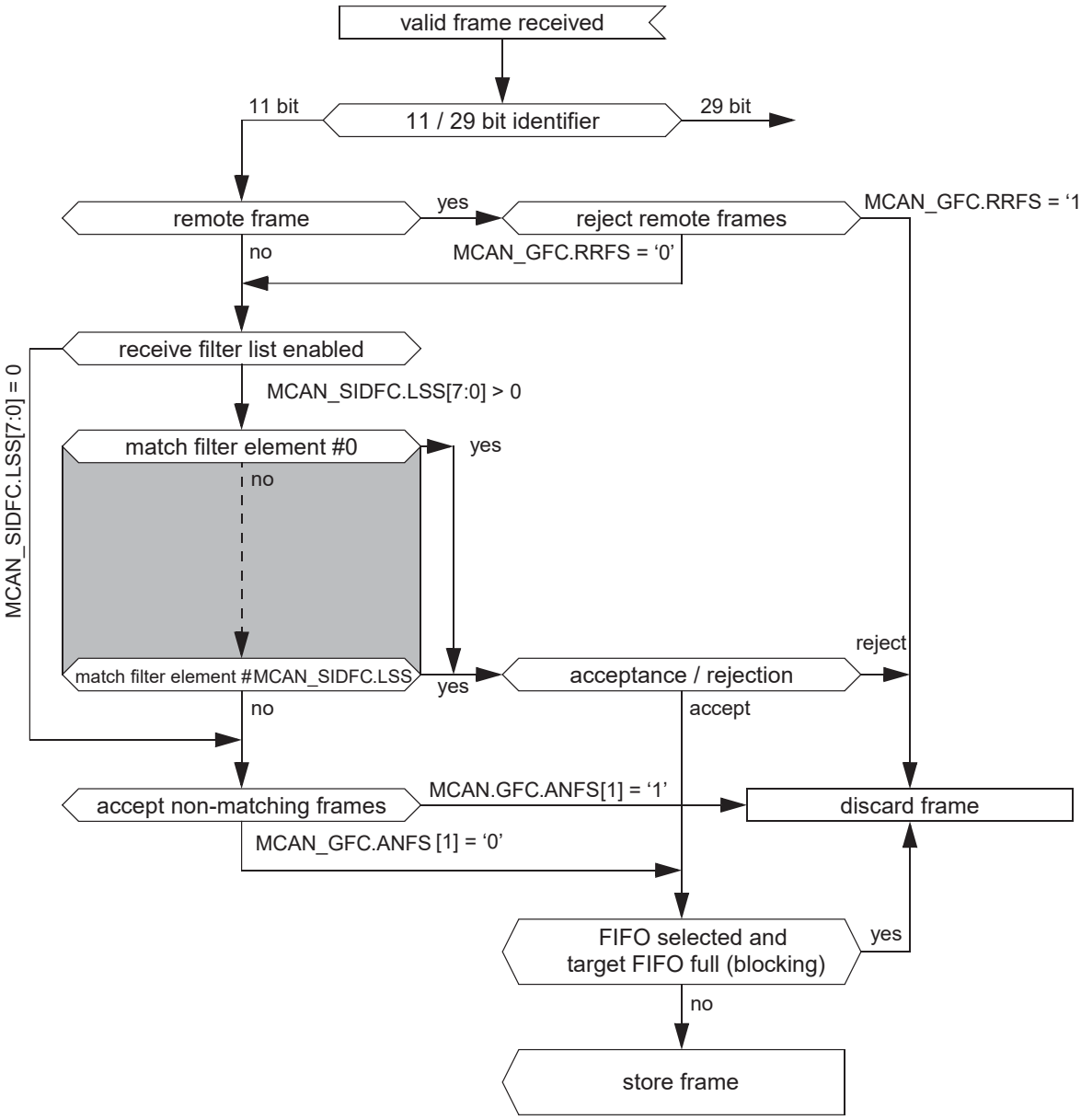
In case all mask bits are one, a match occurs only when the received Message ID and the Message ID filter are identical. If all mask bits are zero, all Message IDs match.

#### 52.5.4.1.4 Standard Message ID Filtering

The figure below shows the flow for standard Message ID (11-bit Identifier) filtering. The Standard Message ID Filter element is described in 52.5.7.5 Standard Message ID Filter Element.

Controlled by MCAN_GFC and MCAN_SIDFC Message ID, Remote Transmission Request bit (RTR), and the Identifier Extension bit (IDE) of received frames are compared against the list of configured filter elements.

**Figure 52-5. Standard Message ID Filter Path**



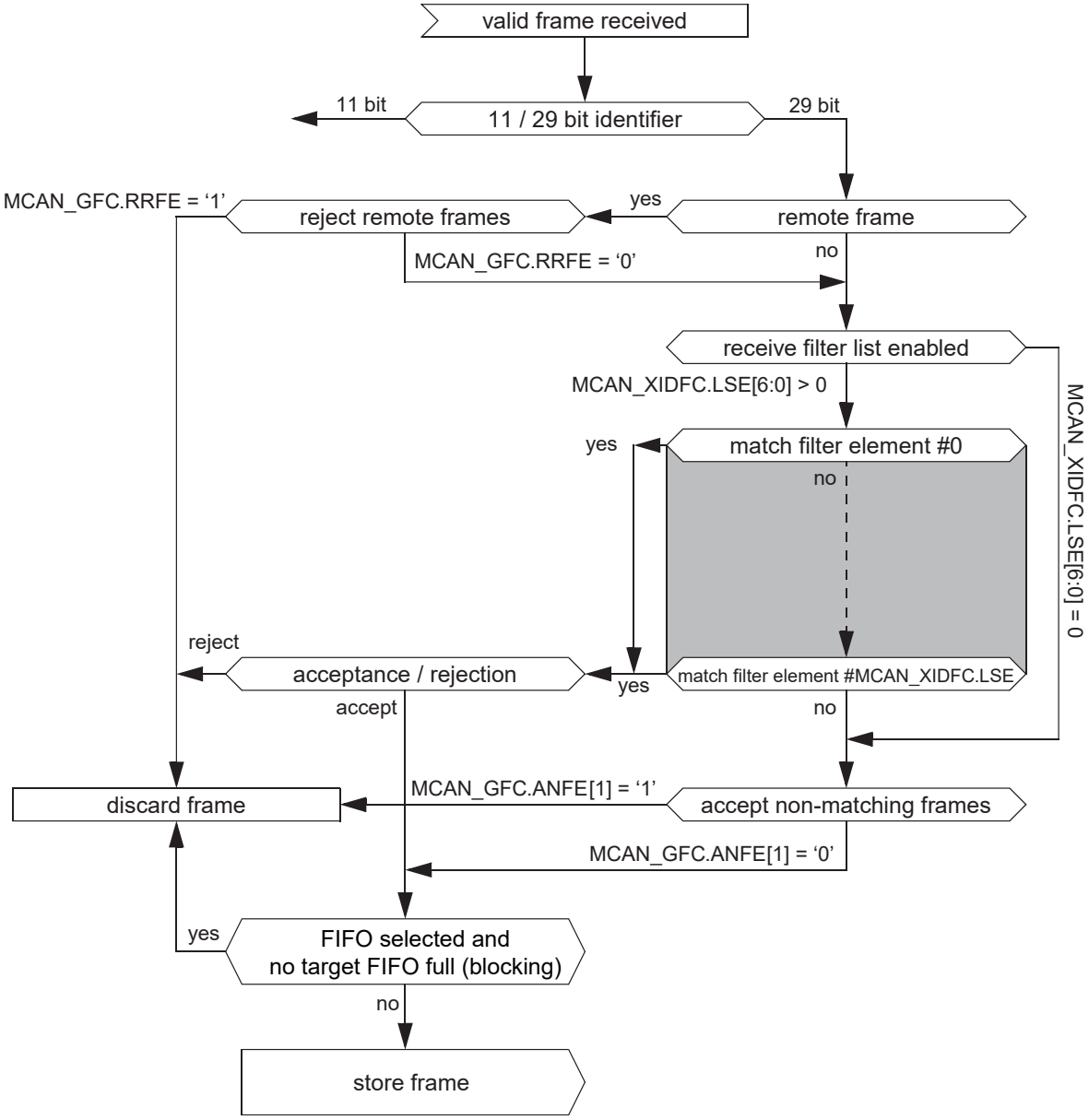

***Extended Message ID Filtering***

The figure below shows the flow for extended Message ID (29-bit Identifier) filtering. The Extended Message ID Filter element is described in 52.5.7.6 Extended Message ID Filter Element.

Controlled by MCAN_GFC and MCAN_XIDFC Message ID, Remote Transmission Request bit (RTR), and the Identifier Extension bit (IDE) of received frames are compared against the list of configured filter elements.

MCAN_XIDAM is ANDed with the received identifier before the filter list is executed.

**Figure 52-6. Extended Message ID Filter Path**



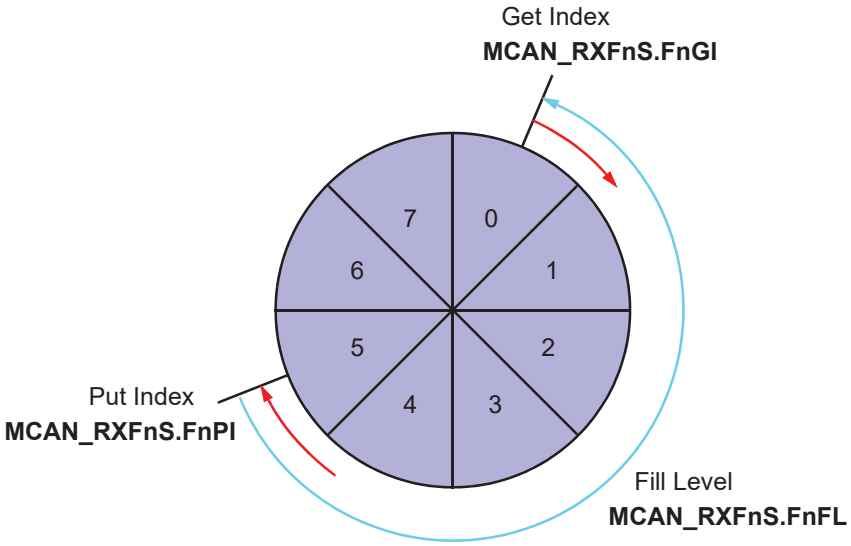

### 52.5.4.2 Rx FIFOs

Rx FIFO 0 and Rx FIFO 1 can be configured to hold up to 64 elements each. Configuration of the two Rx FIFOs is done via the Rx FIFO 0 Configuration register (MCAN_RXF0C) and the Rx FIFO 1 Configuration register (MCAN_RXF1C).

Received messages that passed acceptance filtering are transferred to the Rx FIFO as configured by the matching filter element. For a description of the filter mechanisms available for Rx FIFO 0 and Rx FIFO 1, see Acceptance Filtering. The Rx FIFO element is described in Rx Buffer and FIFO Element.

To avoid an Rx FIFO overflow, the Rx FIFO watermark can be used. When the Rx FIFO fill level reaches the Rx FIFO watermark configured by MCAN_RXFnC.FnWM, interrupt flag MCAN_IR.RFnW is set. When the Rx FIFO Put Index reaches the Rx FIFO Get Index, an Rx FIFO Full condition is signalled by MCAN_RXFnS.FnF. In addition, the interrupt flag MCAN_IR.RFnF is set.

**Figure 52-7. Rx FIFO Status**




When reading from an Rx FIFO, Rx FIFO Get Index MCAN_RXFnS.FnGI × FIFO Element Size has to be added to the corresponding Rx FIFO start address MCAN_RXFnC.FnSA.

**Table 52-2. Rx Buffer / FIFO Element Size**

| MCAN_RXESC.RBDS[2:0]<br>MCAN_RXESC.FnDS[2:0] | Data Field<br>[bytes] | FIFO Element Size<br>[RAM words] |
|---|---|---|
| 0 | 8 | 4 |
| 1 | 12 | 5 |
| 2 | 16 | 6 |
| 3 | 20 | 7 |
| 4 | 24 | 8 |
| 5 | 32 | 10 |
| 6 | 48 | 14 |
| 7 | 64 | 18 |

#### 52.5.4.2.1 Rx FIFO Blocking Mode

The Rx FIFO Blocking mode is configured by MCAN_RXFnC.FnOM = '0'. This is the default operating mode for the Rx FIFOs.

When an Rx FIFO full condition is reached (MCAN_RXFnS.FnPI = MCAN_RXFnS.FnGI), no further messages are written to the corresponding Rx FIFO until at least one message has been read out and the Rx FIFO Get Index has been incremented. An Rx FIFO full condition is signalled by MCAN_RXFnS.FnF = '1'. In addition, the interrupt flag MCAN_IR.RFnF is set.

In case a message is received while the corresponding Rx FIFO is full, this message is discarded and the message lost condition is signalled by MCAN_RXFnS.RFnL = '1'. In addition, the interrupt flag MCAN_IR.RFnL is set.

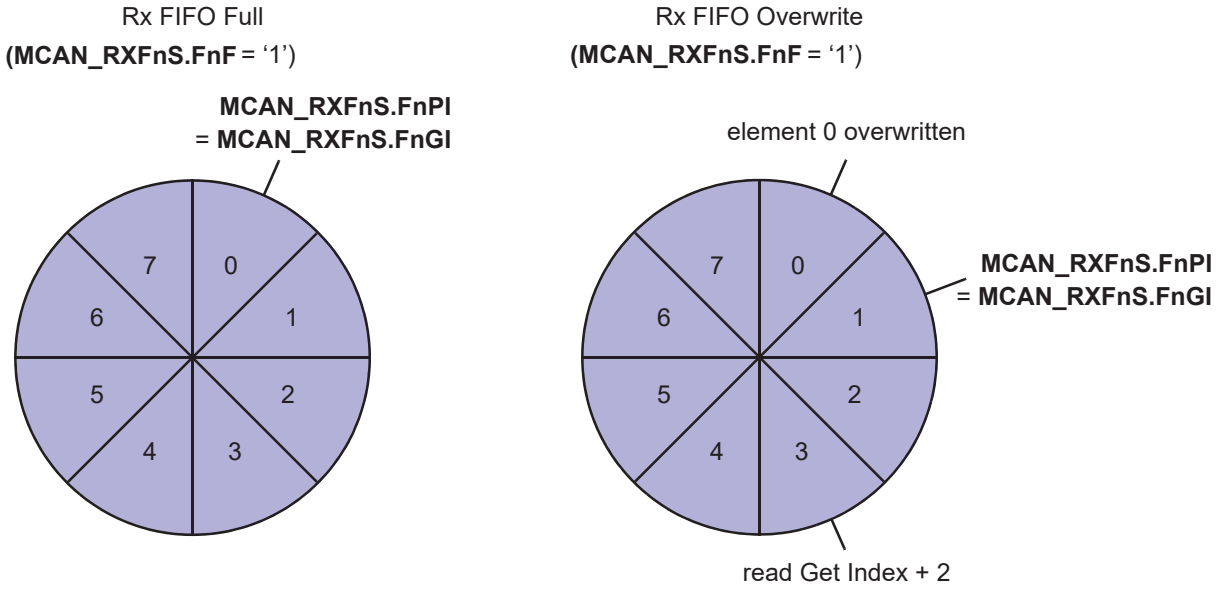
#### 52.5.4.2.2 Rx FIFO Overwrite Mode

The Rx FIFO Overwrite mode is configured by MCAN_RXFnC.FnOM = '1'.

When an Rx FIFO full condition (MCAN_RXFnS.FnPI = MCAN_RXFnS.FnGI) is signalled by MCAN_RXFnS.FnF = '1', the next message accepted for the FIFO will overwrite the oldest FIFO message. Put and get index are both incremented by one.

When an Rx FIFO is operated in Overwrite mode and an Rx FIFO full condition is signalled, reading of the Rx FIFO elements should start at least at get index + 1. The reason for that is, that it might happen, that a received message is written to the Message RAM (put index) while the processor is reading from the Message RAM (get index). In this case inconsistent data may be read from the respective Rx FIFO element. Adding an offset to the get index when reading from the Rx FIFO avoids this problem. The offset depends on how fast the processor accesses the Rx FIFO. The figure below shows an offset of two with respect to the get index when reading the Rx FIFO. In this case the two messages stored in element 1 and 2 are lost.

**Figure 52-8. Rx FIFO Overflow Handling**




After reading from the Rx FIFO, the number of the last element read has to be written to the Rx FIFO Acknowledge Index MCAN_RXFnA.FnA. This increments the get index to that element number. In case the put index has not been incremented to this Rx FIFO element, the Rx FIFO full condition is reset (MCAN_RXFnS.FnF = '0').

### 52.5.4.3 Dedicated Rx Buffers

The MCAN supports up to 64 dedicated Rx Buffers. The start address of the dedicated Rx Buffer section is configured via MCAN_RXBC.RBSA.

For each Rx Buffer, a Standard or Extended Message ID Filter Element with SFEC / EFEC = 7 and SFID2 / EFID2[10:9] = 0 has to be configured.

After a received message has been accepted by a filter element, the message is stored into the Rx Buffer in the Message RAM referenced by the filter element. The format is the same as for an Rx FIFO element. In addition, the flag MCAN_IR.DRX (Message stored in dedicated Rx Buffer) in MCAN_IR is set.

**Table 52-3. Example Filter Configuration for Rx Buffers**

| Filter Element | SFID1[10:0] EFID1[28:0] | SFID2[10:9] EFID2[10:9] | SFID2[5:0] EFID2[5:0] |
|---|---|---|---|
| 0 | ID message 1 | 0 | 0 |

| .....continued | | | |
|---|---|---|---|
| **Filter Element** | **SFID1[10:0] EFID1[28:0]** | **SFID2[10:9] EFID2[10:9]** | **SFID2[5:0] EFID2[5:0]** |
| 1 | ID message 2 | 0 | 1 |
| 2 | ID message 3 | 0 | 2 |

After the last word of a matching received message has been written to the Message RAM, the respective New Data flag in the New Data 1 register (MCAN_NDAT1) and New Data 2 register (MCAN_NDAT2) is set. As long as the New Data flag is set, the respective Rx Buffer is locked against updates from received matching frames. The New Data flags have to be reset by the processor by writing a '1' to the respective bit position.

While an Rx Buffer's New Data flag is set, a Message ID Filter Element referencing this specific Rx Buffer will not match, causing the acceptance filtering to continue. Following Message ID Filter Elements may cause the received message to be stored into another Rx Buffer, or into an Rx FIFO, or the message may be rejected, depending on filter configuration.

#### 52.5.4.3.1 Rx Buffer Handling

- Reset interrupt flag IR.DRX
- Read New Data registers
- Read messages from Message RAM
- Reset New Data flags of processed messages

### 52.5.4.4 Debug on CAN Support

Debug messages are stored into Rx Buffers. For debug handling three consecutive Rx buffers (e.g. #61, #62, #63) have to be used for storage of debug messages A, B, and C. The format is the same as for an Rx Buffer or an Rx FIFO element (see Rx Buffer and FIFO Element).

Advantage: Fixed start address for the DMA transfers (relative to MCAN_RXBC.RBSA), no additional configuration required.

For filtering of debug messages Standard / Extended Filter Elements with SFEC / EFEC = '111' have to be set up. Messages matching these filter elements are stored into the Rx Buffers addressed by SFID2 / EFID2[5:0].

After message C has been stored, the DMA request output m_can_dma_req is activated and the three messages can be read from the Message RAM under DMA control. The RAM words holding the debug messages will not be changed by the MCAN while m_can_dma_req is activated. The behavior is similar to that of an Rx Buffer with its New Data flag set.

After the DMA has completed, the MCAN is prepared to receive the next set of debug messages.

#### 52.5.4.4.1 Filtering for Debug Messages

Filtering for debug messages is done by configuring one Standard / Extended Message ID Filter Element for each of the three debug messages. To enable a filter element to filter for debug messages SFEC / EFEC has to be programmed to "111". In this case fields SFID1 / SFID2 and EFID1 / EFID2 have a different meaning. While SFID2 / EFID2[10:9] controls the debug message handling state machine, SFID2 / EFID2[5:0] controls the location for storage of a received debug message.

When a debug message is stored, neither the respective New Data flag nor MCAN_IR.DRX are set. The reception of debug messages can be monitored via RXF1S.DMS.

**Table 52-4. Example Filter Configuration for Debug Messages**

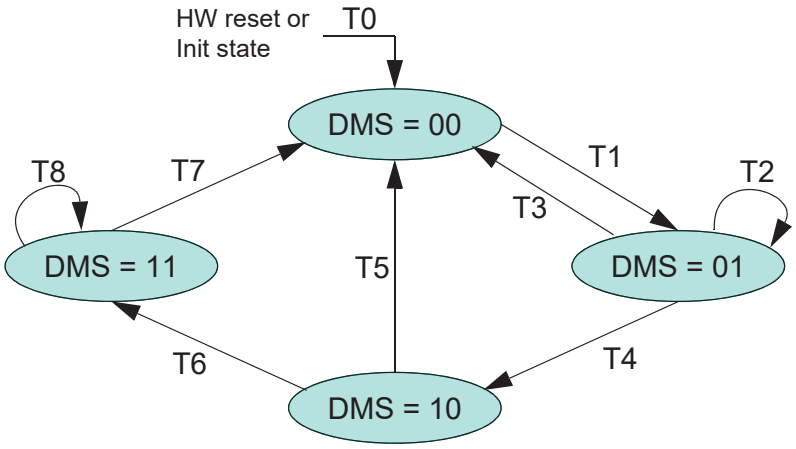
| Filter Element | SFID1[10:0] EFID1[28:0] | SFID2[10:9] EFID2[10:9] | SFID2[5:0] EFID2[5:0] |
|---|---|---|---|
| 0 | ID debug message A | 1 | 11 1101 |
| 1 | ID debug message B | 2 | 11 1110 |
| 2 | ID debug message C | 3 | 11 1111 |

#### 52.5.4.4.2 Debug Message Handling

The debug message handling state machine ensures that debug messages are stored to three consecutive Rx Buffers in the correct order. If some messages are missing, the process is restarted. The DMA request is activated only when all three debug messages A, B, C have been received in the correct order.

The status of the debug message handling state machine is signalled via MCAN_RXF1S.DMS.

**Figure 52-9. Debug Message Handling State Machine**




T0: reset m_can_dma_req output, enable reception of debug messages A, B, and C

T1: reception of debug message A

T2: reception of debug message A

T3: reception of debug message C

T4: reception of debug message B

T5: reception of debug messages A, B

T6: reception of debug message C

T7: DMA transfer completed

T8: reception of debug message A,B,C (message rejected)

### 52.5.5 Tx Handling

The Tx Handler handles transmission requests for the dedicated Tx Buffers, the Tx FIFO, and the Tx Queue. It controls the transfer of transmit messages to the CAN Core, the Put and Get Indices, and the Tx Event FIFO. Up to 32 Tx Buffers can be set up for message transmission. The CAN mode for transmission (Classic CAN or CAN FD) can be configured separately for each Tx Buffer element. The Tx

Buffer element is described in Tx Buffer Element. The table below describes the possible configurations for frame transmission.

**Table 52-5. Possible Configurations for Frame Transmission**

| MCAN_CCCR | | Tx Buffer Element | | Frame Transmission |
|---|---|---|---|---|
| BRSE | FDOE | FDF | BRS | |
| ignored | 0 | ignored | ignored | Classic CAN |
| 0 | 1 | 0 | ignored | Classic CAN |
| 0 | 1 | 1 | ignored | FD without bit rate switching |
| 1 | 1 | 0 | ignored | Classic CAN |
| 1 | 1 | 1 | 0 | FD without bit rate switching |
| 1 | 1 | 1 | 1 | FD with bit rate switching |

**Note:** AUTOSAR requires at least three Tx Queue Buffers and support of transmit cancellation.

The Tx Handler starts a Tx scan to check for the highest priority pending Tx request (Tx Buffer with lowest Message ID) when MCAN_TXBRP is updated, or when a transmission has been started.

#### 52.5.5.1 Transmit Pause

The transmit pause feature is intended for use in CAN systems where the CAN message identifiers are (permanently) specified to specific values and cannot easily be changed. These message identifiers may have a higher CAN arbitration priority than other defined messages, while in a specific application their relative arbitration priority should be inverse. This may lead to a case where one ECU sends a burst of CAN messages that cause another ECU's CAN messages to be delayed because that other messages have a lower CAN arbitration priority.

If e.g. CAN ECU-1 has the transmit pause feature enabled and is requested by its application software to transmit four messages, it will, after the first successful message transmission, wait for two CAN bit times of bus idle before it is allowed to start the next requested message. If there are other ECUs with pending messages, those messages are started in the idle time, they would not need to arbitrate with the next message of ECU-1. After having received a message, ECU-1 is allowed to start its next transmission as soon as the received message releases the CAN bus.

The transmit pause feature is controlled by bit MCAN_CCCR.TXP. If the bit is set, the MCAN will, each time it has successfully transmitted a message, pause for two CAN bit times before starting the next transmission. This enables other CAN nodes in the network to transmit messages even if their messages have lower prior identifiers. Default is transmit pause disabled (MCAN_CCCR.TXP = '0').

This feature looses up burst transmissions coming from a single node and it protects against "babbling idiot" scenarios where the application program erroneously requests too many transmissions.

#### 52.5.5.2 Dedicated Tx Buffers

Dedicated Tx Buffers are intended for message transmission under complete control of the processor. Each dedicated Tx Buffer is configured with a specific Message ID. In case that multiple Tx Buffers are configured with the same Message ID, the Tx Buffer with the lowest buffer number is transmitted first.

If the data section has been updated, a transmission is requested by an Add Request via MCAN_TXBAR.ARn. The requested messages arbitrate internally with messages from an optional Tx FIFO or Tx Queue and externally with messages on the CAN bus, and are sent out according to their Message ID.

A dedicated Tx Buffer allocates Element Size 32-bit words in the Message RAM (see the table below). Therefore the start address of a dedicated Tx Buffer in the Message RAM is calculated by adding transmit buffer index (0…31) × Element Size to the Tx Buffer Start Address TXBC.TBSA.

**Table 52-6. Tx Buffer / FIFO / Queue Element Size**

| TXESC.TBDS[2:0] | Data Field [bytes] | Element Size [RAM words] |
|---|---|---|
| 0 | 8 | 4 |
| 1 | 12 | 5 |
| 2 | 16 | 6 |
| 3 | 20 | 7 |
| 4 | 24 | 8 |
| 5 | 32 | 10 |
| 6 | 48 | 14 |
| 7 | 64 | 18 |

#### 52.5.5.3 Tx FIFO

Tx FIFO operation is configured by programming MCAN_TXBC.TFQM to '0'. Messages stored in the Tx FIFO are transmitted starting with the message referenced by the Get Index MCAN_TXFQS.TFGI. After each transmission the Get Index is incremented cyclically until the Tx FIFO is empty. The Tx FIFO enables transmission of messages with the same Message ID from different Tx Buffers in the order these messages have been written to the Tx FIFO. The MCAN calculates the Tx FIFO Free Level MCAN_TXFQS.TFFL as difference between Get and Put Index. It indicates the number of available (free) Tx FIFO elements.

New transmit messages have to be written to the Tx FIFO starting with the Tx Buffer referenced by the Put Index MCAN_TXFQS.TFQPI. An Add Request increments the Put Index to the next free Tx FIFO element. When the Put Index reaches the Get Index, Tx FIFO Full (MCAN_TXFQS.TFQF = '1') is signalled. In this case no further messages should be written to the Tx FIFO until the next message has been transmitted and the Get Index has been incremented.

When a single message is added to the Tx FIFO, the transmission is requested by writing a '1' to the TXBAR bit related to the Tx Buffer referenced by the Tx FIFO's Put Index.

When multiple (n) messages are added to the Tx FIFO, they are written to n consecutive Tx Buffers starting with the Put Index. The transmissions are then requested via MCAN_TXBAR. The Put Index is then cyclically incremented by n. The number of requested Tx buffers should not exceed the number of free Tx Buffers as indicated by the Tx FIFO Free Level.

When a transmission request for the Tx Buffer referenced by the Get Index is cancelled, the Get Index is incremented to the next Tx Buffer with pending transmission request and the Tx FIFO Free Level is recalculated. When transmission cancellation is applied to any other Tx Buffer, the Get Index and the FIFO Free Level remain unchanged.

A Tx FIFO element allocates Element Size 32-bit words in the Message RAM (see the table Table 52-6). Therefore the start address of the next available (free) Tx FIFO Buffer is calculated by adding Tx FIFO/ Queue Put Index MCAN_TXFQS.TFQPI (0…31) × Element Size to the Tx Buffer Start Address MCAN_TXBC.TBSA.

#### 52.5.5.4 Tx Queue

Tx Queue operation is configured by programming MCAN_TXBC.TFQM to '1'. Messages stored in the Tx Queue are transmitted starting with the message with the lowest Message ID (highest priority). In case that multiple Queue Buffers are configured with the same Message ID, the Queue Buffer with the lowest buffer number is transmitted first.

New messages have to be written to the Tx Buffer referenced by the Put Index MCAN_TXFQS.TFQPI. An Add Request cyclically increments the Put Index to the next free Tx Buffer. In case that the Tx Queue is full (MCAN_TXFQS.TFQF = '1'), the Put Index is not valid and no further message should be written to the Tx Queue until at least one of the requested messages has been sent out or a pending transmission request has been cancelled.

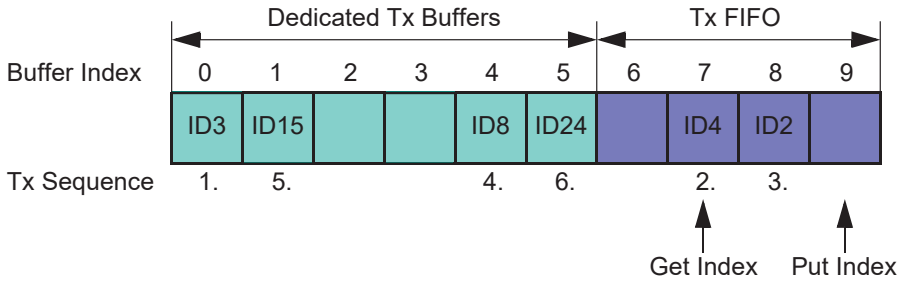
The application may use register MCAN_TXBRP instead of the Put Index and may place messages to any Tx Buffer without pending transmission request.

A Tx Queue Buffer allocates Element Size 32-bit words in the Message RAM (see the table Tx Buffer / FIFO / Queue Element Size). Therefore the start address of the next available (free) Tx Queue Buffer is calculated by adding Tx FIFO/Queue Put Index MCAN_TXFQS.TFQPI (0…31) × Element Size to the Tx Buffer Start Address MCAN_TXBC.TBSA.

#### 52.5.5.5 Mixed Dedicated Tx Buffers / Tx FIFO

In this case the Tx Buffers section in the Message RAM is subdivided into a set of dedicated Tx Buffers and a Tx FIFO. The number of dedicated Tx Buffers is configured by MCAN_TXBC.NDTB. The number of Tx Buffers assigned to the Tx FIFO is configured by MCAN_TXBC.TFQS. In case MCAN_TXBC.TFQS is programmed to zero, only dedicated Tx Buffers are used.

**Figure 52-10. Example of Mixed Configuration Dedicated Tx Buffers / Tx FIFO**



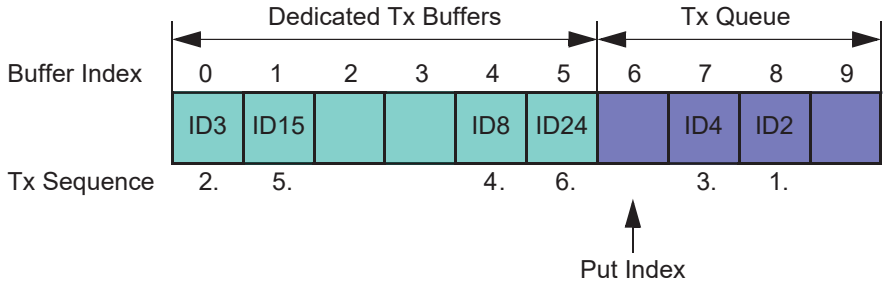

Tx prioritization:

- Scan dedicated Tx Buffers and oldest pending Tx FIFO Buffer (referenced by MCAN_TXFS.TFGI)
- Buffer with lowest Message ID gets highest priority and is transmitted next

#### 52.5.5.6 Mixed Dedicated Tx Buffers / Tx Queue

In this case the Tx Buffers section in the Message RAM is subdivided into a set of dedicated Tx Buffers and a Tx Queue. The number of dedicated Tx Buffers is configured by MCAN_TXBC.NDTB. The number of Tx Queue Buffers is configured by MCAN_TXBC.TFQS. In case MCAN_TXBC.TFQS is programmed to zero, only dedicated Tx Buffers are used.

**Figure 52-11. Example of Mixed Configuration Dedicated Tx Buffers / Tx Queue**




Tx prioritization:

- Scan all Tx Buffers with activated transmission request
- Tx Buffer with lowest Message ID gets highest priority and is transmitted next

### 52.5.5.7 Transmit Cancellation

The MCAN supports transmit cancellation. This feature is especially intended for gateway applications and AUTOSAR-based applications. To cancel a requested transmission from a dedicated Tx Buffer or a Tx Queue Buffer, the processor has to write a '1' to the corresponding bit position (=number of Tx Buffer) of register MCAN_TXBCR. Transmit cancellation is not intended for Tx FIFO operation.

Successful cancellation is signalled by setting the corresponding bit of register MCAN_TXBCF to '1'.

In case a transmit cancellation is requested while a transmission from a Tx Buffer is already ongoing, the corresponding TXBRP bit remains set as long as the transmission is in progress. If the transmission was successful, the corresponding MCAN_TXBTO and MCAN_TXBCF bits are set. If the transmission was not successful, it is not repeated and only the corresponding MCAN_TXBCF bit is set.

**Note:** In case a pending transmission is cancelled immediately before this transmission could have been started, there follows a short time window where no transmission is started even if another message is also pending in this node. This may enable another node to transmit a message which may have a lower priority than the second message in this node.

### 52.5.5.8 Tx Event Handling

To support Tx event handling the MCAN has implemented a Tx Event FIFO. After the MCAN has transmitted a message on the CAN bus, Message ID and timestamp are stored in a Tx Event FIFO element. To link a Tx event to a Tx Event FIFO element, the Message Marker from the transmitted Tx Buffer is copied into the Tx Event FIFO element.

The Tx Event FIFO can be configured to a maximum of 32 elements. The Tx Event FIFO element is described in Debug on CAN Support.

When a Tx Event FIFO full condition is signalled by IR.TEFF, no further elements are written to the Tx Event FIFO until at least one element has been read out and the Tx Event FIFO Get Index has been incremented. In case a Tx event occurs while the Tx Event FIFO is full, this event is discarded and interrupt flag MCAN_IR.TEFL is set.

To avoid a Tx Event FIFO overflow, the Tx Event FIFO watermark can be used. When the Tx Event FIFO fill level reaches the Tx Event FIFO watermark configured by MCAN_TXEFC.EFWM, interrupt flag MCAN_IR.TEFW is set.

When reading from the Tx Event FIFO, two times the Tx Event FIFO Get Index MCAN_TXEFS.EFGI has to be added to the Tx Event FIFO start address MCAN_TXEFC.EFSA.

### 52.5.6 FIFO Acknowledge Handling

The Get Indices of Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO are controlled by writing to the corresponding FIFO Acknowledge Index in the registers MCAN_RXF0A, MCAN_RXF1A and MCAN_TXEFA. Writing to the FIFO Acknowledge Index will set the FIFO Get Index to the FIFO Acknowledge Index plus one and thereby updates the FIFO Fill Level. There are two use cases:

When only a single element has been read from the FIFO (the one being pointed to by the Get Index), this Get Index value is written to the FIFO Acknowledge Index.

When a sequence of elements has been read from the FIFO, it is sufficient to write the FIFO Acknowledge Index only once at the end of that read sequence (value: Index of the last element read), to update the FIFO's Get Index.

Due to the fact that the processor has free access to the MCAN's Message RAM, special care has to be taken when reading FIFO elements in an arbitrary order (Get Index not considered). This might be useful when reading a High Priority Message from one of the two Rx FIFOs. In this case the FIFO's Acknowledge Index should not be written because this would set the Get Index to a wrong position and also alters the FIFO's Fill Level. In this case some of the older FIFO elements would be lost.

**Note:** The application has to ensure that a valid value is written to the FIFO Acknowledge Index. The MCAN does not check for erroneous values.

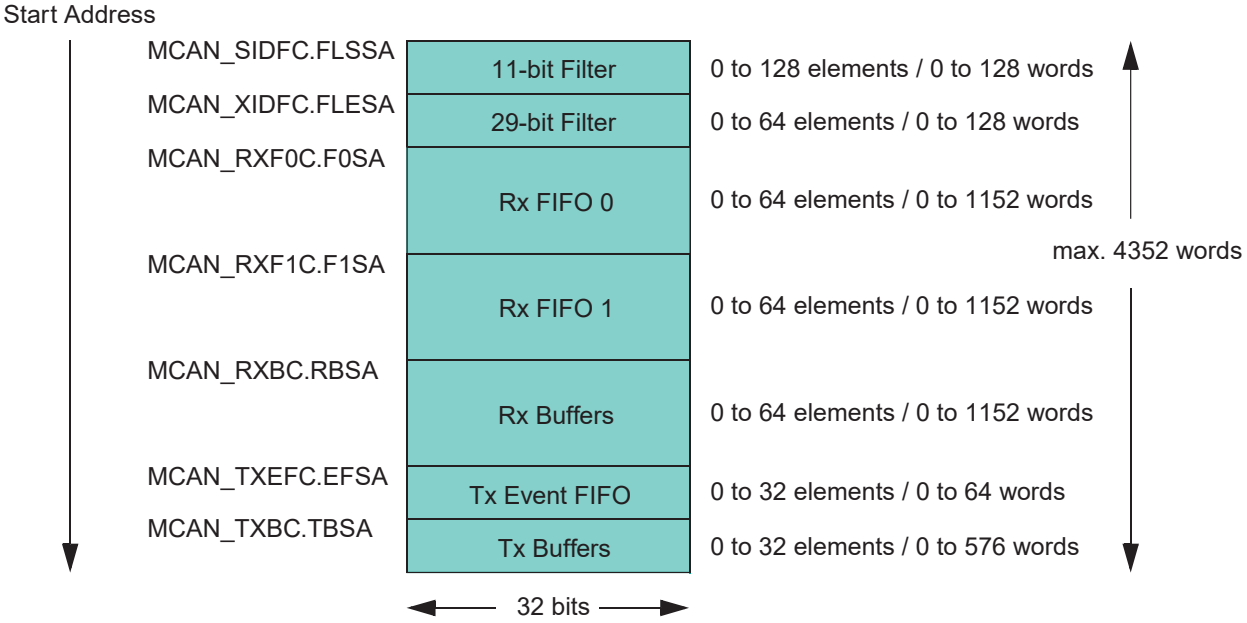
### 52.5.7 Message RAM

#### 52.5.7.1 Message RAM Configuration

The Message RAM has a width of 32 bits. The MCAN module can be configured to allocate up to 4352 words in the Message RAM. It is not necessary to configure each of the sections listed in the figure below, nor is there any restriction with respect to the sequence of the sections.

When operated in CAN FD mode, the required Message RAM size depends on the element size configured for Rx FIFO0, Rx FIFO1, Rx Buffers, and Tx Buffers via MCAN_RXESC.F0DS, MCAN_RXESC.F1DS, MCAN_RXESC.RBDS, and MCAN_TXESC.TBDS.

**Figure 52-12. Message RAM Configuration**



When the MCAN addresses the Message RAM, it addresses 32-bit words, not single bytes. The configurable start addresses are 32-bit word addresses; i.e., only bits 15 to 2 are evaluated, the two least significant bits are ignored.

**Note:** The MCAN does not check for erroneous configuration of the Message RAM. The configuration of the start addresses of the different sections and the number of elements of each section must be checked carefully to avoid falsification or loss of data.

#### 52.5.7.2 Rx Buffer and FIFO Element

Up to 64 Rx Buffers and two Rx FIFOs can be configured in the Message RAM. Each Rx FIFO section can be configured to store up to 64 received messages. The structure of a Rx Buffer / FIFO element is shown in the table below. The element size can be configured for storage of CAN FD messages with up to 64 bytes data field via register MCAN_RXESC.

**Table 52-7. Rx Buffer and FIFO Element**

| | 31 | | | 24 | 23 | | | | 16 | 15 | | 8 | 7 | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R0 | ESI | XTD | RTR | ID[28:0] | | | | | | | | | | | |
| R1 | ANMF | FIDX[6:0] | | | – | FDF | BRS | DLC[3:0] | | RXTS[15:0] | | | | | |
| R2 | DB3[7:0] | | | | DB2[7:0] | | | | | DB1[7:0] | | | DB0[7:0] | | |
| R3 | DB7[7:0] | | | | DB6[7:0] | | | | | DB5[7:0] | | | DB4[7:0] | | |
| ... | ... | | | | ... | | | | | ... | | | ... | | |
| Rn | DBm[7:0] | | | | DBm-1[7:0] | | | | | DBm-2[7:0] | | | DBm-3[7:0] | | |

• R0 Bit 31 ESI: Error State Indicator

0: Transmitting node is error active.

1: Transmitting node is error passive.

• R0 Bit 30 XTD: Extended Identifier

Signals to the processor whether the received frame has a standard or extended identifier.

0: 11-bit standard identifier.

1: 29-bit extended identifier.

• R0 Bit 29 RTR: Remote Transmission Request

Signals to the processor whether the received frame is a data frame or a remote frame.

0: Received frame is a data frame.

1: Received frame is a remote frame.

**Note:** There are no remote frames in CAN FD format. In case a CAN FD frame was received (FDF = 1), bit RTR reflects the state of the reserved bit r1.

• R0 Bits 28:0 ID[28:0]: Identifier

Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].

• R1 Bit 31 ANMF: Accepted Non-matching Frame

Acceptance of non-matching frames may be enabled via MCAN_GFC.ANFS and MCAN_GFC.ANFE.

0: Received frame matching filter index FIDX.

1: Received frame did not match any Rx filter element.

• R1 Bits 30:24 FIDX[6:0]: Filter Index

0-127: Index of matching Rx acceptance filter element (invalid if ANMF = '1').
Range is 0 to MCAN_SIDFC.LSS - 1 resp. MCAN_XIDFC.LSE - 1.

• R1 Bit 21 FDF: FD Format

0: Standard frame format.

1: CAN FD frame format (new DLC-coding and CRC).

• R1 Bit 20 BRS: Bit Rate Switch

0: Frame received without bit rate switching.

1: Frame received with bit rate switching.

**Note:**
Bits ESI, FDF, and BRS are only evaluated when CAN FD operation is enabled (MCAN_CCCR.FDOE = 1). Bit BRS is only evaluated when in addition MCAN_CCCR.BRSE = 1.

• R1 Bits 19:16 DLC[3:0]: Data Length Code

0-8: CAN + CAN FD: received frame has 0-8 data bytes.

9-15: CAN: received frame has 8 data bytes.

9-15: CAN FD: received frame has 12/16/20/24/32/48/64 data bytes.

• R1 Bits 15:0 RXTS[15:0]: Rx Timestamp

Timestamp Counter value captured on start of frame reception. Resolution depending on configuration of the Timestamp Counter Prescaler MCAN_TSCC.TCP.

• R2 Bits 31:24 DB3[7:0]: Data Byte 3

• R2 Bits 23:16 DB2[7:0]: Data Byte 2

• R2 Bits 15:8 DB1[7:0]: Data Byte 1

• R2 Bits 7:0 DB0[7:0]: Data Byte 0

• R3 Bits 31:24 DB7[7:0]: Data Byte 7

• R3 Bits 23:16 DB6[7:0]: Data Byte 6

• R3 Bits 15:8 DB5[7:0]: Data Byte 5

• R3 Bits 7:0 DB4[7:0]: Data Byte 4

... ... ...

• Rn Bits 31:24 DBm[7:0]: Data Byte m

• Rn Bits 23:16 DBm-1[7:0]: Data Byte m-1

• Rn Bits 15:8 DBm-2[7:0]: Data Byte m-2

• Rn Bits 7:0 DBm-3[7:0]: Data Byte m-3

**Note:** Depending on the configuration of the element size (MCAN_RXESC), between two and sixteen 32-bit words (Rn = 3 ..17) are used for storage of a CAN message's data field.

### 52.5.7.3 Tx Buffer Element

The Tx Buffers section can be configured to hold dedicated Tx Buffers as well as a Tx FIFO / Tx Queue. In case that the Tx Buffers section is shared by dedicated Tx buffers and a Tx FIFO / Tx Queue, the dedicated Tx Buffers start at the beginning of the Tx Buffers section followed by the buffers assigned to the Tx FIFO or Tx Queue. The Tx Handler distinguishes between dedicated Tx Buffers and Tx FIFO / Tx Queue by evaluating the Tx Buffer configuration TXBC.TFQS and TXBC.NDTB. The element size can be configured for storage of CAN FD messages with up to 64 bytes data field via register TXESC.

**Table 52-8. Tx Buffer Element**

| | 31 | | | 24 | 23 | | | | 16 | 15 | 8 | 7 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| T0 | ESI | XTD | RTR | ID[28:0] | | | | | | | | | |
| T1 | MM[7:0] | | | | EFC | reserved | FDF | BRS | DLC[3:0] | reserved | | | |
| T2 | DB3[7:0] | | | | DB2[7:0] | | | | | DB1[7:0] | | DB0[7:0] | |
| T3 | DB7[7:0] | | | | DB6[7:0] | | | | | DB5[7:0] | | DB4[7:0] | |
| ... | ... | | | | ... | | | | | ... | | ... | |
| Tn | DBm[7:0] | | | | DBm-1[7:0] | | | | | DBm-2[7:0] | | DBm-3[7:0] | |

• T0 Bit 30 ESI: Error State Indicator

T0 Bit 31 ESI: Error State Indicator

0: ESI bit in CAN FD format depends only on error passive flag

1: ESI bit in CAN FD format transmitted recessive

**Note:** The ESI bit of the transmit buffer is or'ed with the error passive flag to decide the value of the ESI bit in the transmitted FD frame. As required by the CAN FD protocol specification, an error active node may optionally transmit the ESI bit recessive, but an error passive node will always transmit the ESI bit recessive. This feature can be used in gateway applications when a message from an error passive node is routed to another CAN network.

• T0 Bit 30 XTD: Extended Identifier

0: 11-bit standard identifier.

1: 29-bit extended identifier.

• T0 Bit 29 RTR: Remote Transmission Request

0: Transmit data frame.

1: Transmit remote frame.

**Note:** When RTR = 1, the MCAN transmits a remote frame according to ISO11898-1, even if MCAN_CCCR.FDOE enables the transmission in CAN FD format.

• T0 Bits 28:0 ID[28:0]: Identifier

Standard or extended identifier depending on bit XTD. A standard identifier has to be written to ID[28:18].

• T1 Bits 31:24 MM[7:0]: Message Marker

Written by processor during Tx Buffer configuration. Copied into Tx Event FIFO element for identification of Tx message status.

• T1 Bit 23 EFC: Event FIFO Control

0: Do not store Tx events.

1: Store Tx events.

• T1 Bit 21 FDF: FD Format

0: Frame transmitted in Classic CAN format

1: Frame transmitted in CAN FD format

• T1 Bit 20 BRS: Bit Rate Switching

0: CAN FD frames transmitted without bit rate switching

1: CAN FD frames transmitted with bit rate switching

**Note:**
Bits ESI, FDF, and BRS are only evaluated when CAN FD operation is enabled (MCAN_CCCR.FDOE = 1). Bit BRS is only evaluated when in addition MCAN_CCCR.BRSE = 1.

• T1 Bits 19:16 DLC[3:0]: Data Length Code

0-8: CAN + CAN FD: transmit frame has 0-8 data bytes.

9-15: CAN: transmit frame has 8 data bytes.

9-15: CAN FD: transmit frame has 12/16/20/24/32/48/64 data bytes.

• T2 Bits 31:24 DB3[7:0]: Data Byte 3

• T2 Bits 23:16 DB2[7:0]: Data Byte 2

• T2 Bits 15:8 DB1[7:0]: Data Byte 1

• T2 Bits 7:0 DB0[7:0]: Data Byte 0

• T3 Bits 31:24 DB7[7:0]: Data Byte 7

• T3 Bits 23:16 DB6[7:0]: Data Byte 6

• T3 Bits 15:8 DB5[7:0]: Data Byte 5

• T3 Bits 7:0 DB4[7:0]: Data Byte 4

... ... ...

• Tn Bits 31:24 DBm[7:0]: Data Byte m

• Tn Bits 23:16 DBm-1[7:0]: Data Byte m-1

• Tn Bits 15:8 DBm-2[7:0]: Data Byte m-2

• Tn Bits 7:0 DBm-3[7:0]: Data Byte m-3

**Note:** Depending on the configuration of the element size (MCAN_TXESC), between two and sixteen 32-bit words (Tn = 3 ..17) are used for storage of a CAN message's data field.

#### 52.5.7.4 Tx Event FIFO Element

Each element stores information about transmitted messages. By reading the Tx Event FIFO the processor gets this information in the order the messages were transmitted. Status information about the Tx Event FIFO can be obtained from register TXEFS.

**Table 52-9. Tx Event FIFO Element**

| | 31 | | | 24 | 23 | | | | | | 16 | 15 | 8 | 7 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| E0 | ESI | XTD | RTR | ID[28:0] | | | | | | | | | | | |
| E1 | MM[7:0] | | | | ET [1:0] | FDF | BRS | DLC[3:0] | | | | TXTS[15:0] | | | |

• E0 Bit 31 ESI: Error State Indicator

0: Transmitting node is error active.

1: Transmitting node is error passive.

• E0 Bit 30 XTD: Extended Identifier

0: 11-bit standard identifier.

1: 29-bit extended identifier.

• E0 Bit 29 RTR: Remote Transmission Request

0: Data frame transmitted.

1: Remote frame transmitted.

• E0 Bits 28:0 ID[28:0]: Identifier

Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].

• E1 Bits 31:24 MM[7:0]: Message Marker

Copied from Tx Buffer into Tx Event FIFO element for identification of Tx message status.

• E1 Bit 23:22 ET[1:0]: Event Type

0: Reserved

1: Tx event

2: Transmission in spite of cancellation (always set for transmissions in DAR mode)

3: Reserved

• E1 Bit 21 FDF: FD Format

0: Standard frame format.

1: CAN FD frame format (new DLC-coding and CRC).

• E1 Bit 20 BRS: Bit Rate Switch

0: Frame transmitted without bit rate switching.

1: Frame transmitted with bit rate switching.

• E1 Bits 19:16 DLC[3:0]: Data Length Code

0-8: CAN + CAN FD: frame with 0-8 data bytes transmitted.

9-15: CAN: frame with 8 data bytes transmitted.

9-15: CAN FD: frame with 12/16/20/24/32/48/64 data bytes transmitted

• E1 Bits 15:0 TXTS[15:0]: Tx Timestamp

Timestamp Counter value captured on start of frame transmission. Resolution depending on configuration of the Timestamp Counter Prescaler MCAN_TSCC.TCP.

#### 52.5.7.5 Standard Message ID Filter Element

Up to 128 filter elements can be configured for 11-bit standard IDs. When accessing a Standard Message ID Filter element, its address is the Filter List Standard Start Address MCAN_SIDFC.FLSSA plus the index of the filter element (0…127).

**Table 52-10. Standard Message ID Filter Element**

| | 31 | | | | | 24 | 23 | 16 | 15 | 8 | 7 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| S0 | SFT[1:0] | | SFEC [2:0] | | | SFID1[10:0] | | | – | SFID2[10:0] | | |

• Bits 31:30 SFT[1:0]: Standard Filter Type

0: Range filter from SF1ID to SF2ID (SF2ID ≥ SF1ID)

1: Dual ID filter for SF1ID or SF2ID

2: Classic filter: SF1ID = filter, SF2ID = mask

3: Reserved

• Bit 29:27 SFEC[2:0]: Standard Filter Element Configuration

All enabled filter elements are used for acceptance filtering of standard frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If SFEC = “100”, “101”, or “110” a match sets interrupt flag MCAN_IR.HPM and, if enabled, an interrupt is generated. In this case register HPMS is updated with the status of the priority match.

0: Disable filter element

1: Store in Rx FIFO 0 if filter matches

2: Store in Rx FIFO 1 if filter matches

3: Reject ID if filter matches

4: Set priority if filter matches

5: Set priority and store in FIFO 0 if filter matches

6: Set priority and store in FIFO 1 if filter matches

7: Store into Rx Buffer or as debug message, configuration of SFT[1:0] ignored

• Bits 26:16 SFID1[10:0]: Standard Filter ID 1

First ID of standard ID filter element.

When filtering for Rx Buffers or for debug messages this field defines the ID of a standard message to be stored. The received identifiers must match exactly, no masking mechanism is used.

• Bits 10:0 SFID2[10:0]: Standard Filter ID 2

This field has a different meaning depending on the configuration of SFEC:

- SFEC = “001”...“110”–Second ID of standard ID filter element
- SFEC = “111”–Filter for Rx Buffers or for debug messages

SFID2[10:9] decides whether the received message is stored into an Rx Buffer or treated as message A, B, or C of the debug message sequence.

0: Store message in a Rx buffer

1: Debug Message A

2: Debug Message B

3: Debug Message C

SFID2[5:0] defines the index of the dedicated Rx Buffer element to which a matching message is stored.

#### 52.5.7.6 Extended Message ID Filter Element

Up to 64 filter elements can be configured for 29-bit extended IDs. When accessing an Extended Message ID Filter element, its address is the Filter List Extended Start Address MCAN_XIDFC.FLESA plus two times the index of the filter element (0…63).

**Table 52-11. Extended Message ID Filter Element**

| | 31 | | | 24 | 23 | 16 | 15 | 8 | 7 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| F0 | EFEC [2:0] | | | EFID1[28:0] | | | | | | |
| F1 | EFT[1:0] | | – | EFID2[28:0] | | | | | | |

• F0 Bit 31:29 EFEC[2:0]: Extended Filter Element Configuration

All enabled filter elements are used for acceptance filtering of extended frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If EFEC = “100”, “101”, or “110”, a match sets the interrupt flag MCAN_IR.HPM and, if enabled, an interrupt is generated. In this case, register MCAN_HPMS is updated with the status of the priority match.

0: Disable filter element

1: Store in Rx FIFO 0 if filter matches

2: Store in Rx FIFO 1 if filter matches

3: Reject ID if filter matches

4: Set priority if filter matches

5: Set priority and store in FIFO 0 if filter matches

6: Set priority and store in FIFO 1 if filter matches

7: Store into Rx Buffer or as debug message, configuration of EFT[1:0] ignored

• F0 Bits 28:0 EFID1[28:0]: Extended Filter ID 1

First ID of extended ID filter element.

When filtering for Rx Buffers or for debug messages this field defines the ID of an extended message to be stored. The received identifiers must match exactly, only MCAN_XIDAM masking mechanism (see Extended Message ID Filtering) is used.

• F1 Bits 31:30 EFT[1:0]: Extended Filter Type

0: Range filter from EF1ID to EF2ID (EF2ID ≥ EF1ID)

1: Dual ID filter for EF1ID or EF2ID

2: Classic filter: EF1ID = filter, EF2ID = mask

3: Range filter from EF1ID to EF2ID (EF2ID ≥ EF1ID), MCAN_XIDAM mask not applied

• F1 Bits 28:0 EFID2[28:0]: Extended Filter ID 2

This field has a different meaning depending on the configuration of EFEC:

• EFEC = “001”...“110”–Second ID of extended ID filter element

• EFEC = “111”–Filter for Rx Buffers or for debug messages

EFID2[10:9] decides whether the received message is stored into an Rx Buffer or treated as message A, B, or C of the debug message sequence.

0: Store message in an Rx buffer

1: Debug Message A

2: Debug Message B

3: Debug Message C

EFID2[5:0] defines the index of the dedicated Rx Buffer element to which a matching message is stored.

### 52.5.8 Hardware Reset Description

After hardware reset, the registers of the MCAN hold the reset values listed in the register descriptions. Additionally the Bus_Off state is reset and the output CANTX is set to recessive (HIGH). The value 0x0001 (MCAN_CCCR.INIT = ‘1’) in the CC Control register enables software initialization. The MCAN does not influence the CAN bus until the processor resets MCAN_CCCR.INIT to ‘0’.

### 52.5.9 Access to Reserved Register Addresses

In case the application software accesses one of the reserved addresses in the MCAN register map (read or write access), interrupt flag MCAN_IR.ARA is set and, if enabled, the selected interrupt line is risen.

## 52.6 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x04 | MCAN_ENDN | 7:0 | ETV[7:0] | | | | | | | |
| | | 15:8 | ETV[15:8] | | | | | | | |
| | | 23:16 | ETV[23:16] | | | | | | | |
| | | 31:24 | ETV[31:24] | | | | | | | |
| 0x08 | MCAN_CUST | 7:0 | CSV[7:0] | | | | | | | |
| | | 15:8 | CSV[15:8] | | | | | | | |
| | | 23:16 | CSV[23:16] | | | | | | | |
| | | 31:24 | CSV[31:24] | | | | | | | |
| 0x0C | MCAN_DBTP | 7:0 | DTSEG2[3:0] | | | | | DSJW[2:0] | | |
| | | 15:8 | | | | DTSEG1[4:0] | | | | |
| | | 23:16 | TDC | | | DBRP[4:0] | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | MCAN_TEST | 7:0 | RX | TX[1:0] | | LBCK | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | MCAN_RWD | 7:0 | WDC[7:0] | | | | | | | |
| | | 15:8 | WDV[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | MCAN_CCCR | 7:0 | TEST | DAR | MON | CSR | CSA | ASM | CCE | INIT |
| | | 15:8 | | TXP | EFBI | PXHD | | | BRSE | FDOE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | MCAN_NBTP | 7:0 | | NTSEG2[6:0] | | | | | | |
| | | 15:8 | NTSEG1[7:0] | | | | | | | |
| | | 23:16 | NBRP[7:0] | | | | | | | |
| | | 31:24 | NSJW[6:0] | | | | | | | NBRP[8:8] |
| 0x20 | MCAN_TSCC | 7:0 | | | | | | | TSS[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | TCP[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x24 | MCAN_TSCV | 7:0 | TSC[7:0] | | | | | | | |
| | | 15:8 | TSC[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x28 | MCAN_TOCC | 7:0 | | | | | | TOS[1:0] | | ETOC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | TOP[7:0] | | | | | | | |
| | | 31:24 | TOP[15:8] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x2C | MCAN_TOCV | 7:0 | TOC[7:0] | | | | | | | |
| | | 15:8 | TOC[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x30 ... 0x3F | Reserved | | | | | | | | | |
| 0x40 | MCAN_ECR | 7:0 | TEC[7:0] | | | | | | | |
| | | 15:8 | RP | REC[6:0] | | | | | | |
| | | 23:16 | CEL[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x44 | MCAN_PSR | 7:0 | BO | EW | EP | ACT[1:0] | | LEC[2:0] | | |
| | | 15:8 | | PXE | RFDF | RBRS | RESI | DLEC[2:0] | | |
| | | 23:16 | | TDCV[6:0] | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x48 | MCAN_TDCR | 7:0 | | TDCF[6:0] | | | | | | |
| | | 15:8 | | TDCO[6:0] | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x4C ... 0x4F | Reserved | | | | | | | | | |
| 0x50 | MCAN_IR | 7:0 | RF1L | RF1F | RF1W | RF1N | RF0L | RF0F | RF0W | RF0N |
| | | 15:8 | TEFL | TEFF | TEFW | TEFN | TFE | TCF | TC | HPM |
| | | 23:16 | EP | ELO | | | DRX | TOO | MRAF | TSW |
| | | 31:24 | | | ARA | PED | PEA | WDI | BO | EW |
| 0x54 | MCAN_IE | 7:0 | RF1LE | RF1FE | RF1WE | RF1NE | RF0LE | RF0FE | RF0WE | RF0NE |
| | | 15:8 | TEFLE | TEFFE | TEFWE | TEFNE | TFEE | TCFE | TCE | HPME |
| | | 23:16 | EPE | ELOE | | | DRXE | TOOE | MRAFE | TSWE |
| | | 31:24 | | | ARAE | PEDE | PEAE | WDIE | BOE | EWE |
| 0x58 | MCAN_ILS | 7:0 | RF1LL | RF1FL | RF1WL | RF1NL | RF0LL | RF0FL | RF0WL | RF0NL |
| | | 15:8 | TEFLL | TEFFL | TEFWL | TEFNL | TFEL | TCFL | TCL | HPML |
| | | 23:16 | EPL | ELOL | | | DRXL | TOOL | MRAFL | TSWL |
| | | 31:24 | | | ARAL | PEDL | PEAL | WDIL | BOL | EWL |
| 0x5C | MCAN_ILE | 7:0 | | | | | | | EINT1 | EINT0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x60 ... 0x7F | Reserved | | | | | | | | | |
| 0x80 | MCAN_GFC | 7:0 | | | ANFS[1:0] | | ANFE[1:0] | | RRFS | RRFE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x84 | MCAN_SIDFC | 7:0 | FLSSA[5:0] | | | | | | | |
| | | 15:8 | FLSSA[13:6] | | | | | | | |
| | | 23:16 | LSS[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x88 | MCAN_XIDFC | 7:0 | FLESA[5:0] | | | | | | | |
| | | 15:8 | FLESA[13:6] | | | | | | | |
| | | 23:16 | | LSE[6:0] | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x8C ... 0x8F | Reserved | | | | | | | | | |
| 0x90 | MCAN_XIDAM | 7:0 | EIDM[7:0] | | | | | | | |
| | | 15:8 | EIDM[15:8] | | | | | | | |
| | | 23:16 | EIDM[23:16] | | | | | | | |
| | | 31:24 | | | | EIDM[28:24] | | | | |
| 0x94 | MCAN_HPMS | 7:0 | MSI[1:0] | | BIDX[5:0] | | | | | |
| | | 15:8 | FLST | FIDX[6:0] | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x98 | MCAN_NDAT1 | 7:0 | ND7 | ND6 | ND5 | ND4 | ND3 | ND2 | ND1 | ND0 |
| | | 15:8 | ND15 | ND14 | ND13 | ND12 | ND11 | ND10 | ND9 | ND8 |
| | | 23:16 | ND23 | ND22 | ND21 | ND20 | ND19 | ND18 | ND17 | ND16 |
| | | 31:24 | ND31 | ND30 | ND29 | ND28 | ND27 | ND26 | ND25 | ND24 |
| 0x9C | MCAN_NDAT2 | 7:0 | ND39 | ND38 | ND37 | ND36 | ND35 | ND34 | ND33 | ND32 |
| | | 15:8 | ND47 | ND46 | ND45 | ND44 | ND43 | ND42 | ND41 | ND40 |
| | | 23:16 | ND55 | ND54 | ND53 | ND52 | ND51 | ND50 | ND49 | ND48 |
| | | 31:24 | ND63 | ND62 | ND61 | ND60 | ND59 | ND58 | ND57 | ND56 |
| 0xA0 | MCAN_RXF0C | 7:0 | F0SA[5:0] | | | | | | | |
| | | 15:8 | F0SA[13:6] | | | | | | | |
| | | 23:16 | | F0S[6:0] | | | | | | |
| | | 31:24 | F0OM | F0WM[6:0] | | | | | | |
| 0xA4 | MCAN_RXF0S | 7:0 | | F0FL[6:0] | | | | | | |
| | | 15:8 | | | F0GI[5:0] | | | | | |
| | | 23:16 | | | F0PI[5:0] | | | | | |
| | | 31:24 | | | | | | | RF0L | F0F |
| 0xA8 | MCAN_RXF0A | 7:0 | | | F0AI[5:0] | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xAC | MCAN_RXBC | 7:0 | RBSA[5:0] | | | | | | | |
| | | 15:8 | RBSA[13:6] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xB0 | MCAN_RXF1C | 7:0 | F1SA[5:0] | | | | | | | |
| | | 15:8 | F1SA[13:6] | | | | | | | |
| | | 23:16 | | F1S[6:0] | | | | | | |
| | | 31:24 | F1OM | F1WM[6:0] | | | | | | |
| 0xB4 | MCAN_RXF1S | 7:0 | | F1FL[6:0] | | | | | | |
| | | 15:8 | | | F1GI[5:0] | | | | | |
| | | 23:16 | | | F1PI[5:0] | | | | | |
| | | 31:24 | DMS[1:0] | | | | | | RF1L | F1F |
| 0xB8 | MCAN_RXF1A | 7:0 | | | F1AI[5:0] | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xBC | MCAN_RXESC | 7:0 | | F1DS[2:0] | | | | F0DS[2:0] | | |
| | | 15:8 | | | | | | RBDS[2:0] | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xC0 | MCAN_TXBC | 7:0 | TBSA[5:0] | | | | | | | |
| | | 15:8 | TBSA[13:6] | | | | | | | |
| | | 23:16 | | | NDTB[5:0] | | | | | |
| | | 31:24 | | TFQM | TFQS[5:0] | | | | | |
| 0xC4 | MCAN_TXFQS | 7:0 | | | TFFL[5:0] | | | | | |
| | | 15:8 | | | | TFGI[4:0] | | | | |
| | | 23:16 | | | TFQF | TFQPI[4:0] | | | | |
| | | 31:24 | | | | | | | | |
| 0xC8 | MCAN_TXESC | 7:0 | | | | | | TBDS[2:0] | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xCC | MCAN_TXBRP | 7:0 | TRP7 | TRP6 | TRP5 | TRP4 | TRP3 | TRP2 | TRP1 | TRP0 |
| | | 15:8 | TRP15 | TRP14 | TRP13 | TRP12 | TRP11 | TRP10 | TRP9 | TRP8 |
| | | 23:16 | TRP23 | TRP22 | TRP21 | TRP20 | TRP19 | TRP18 | TRP17 | TRP16 |
| | | 31:24 | TRP31 | TRP30 | TRP29 | TRP28 | TRP27 | TRP26 | TRP25 | TRP24 |
| 0xD0 | MCAN_TXBAR | 7:0 | AR7 | AR6 | AR5 | AR4 | AR3 | AR2 | AR1 | AR0 |
| | | 15:8 | AR15 | AR14 | AR13 | AR12 | AR11 | AR10 | AR9 | AR8 |
| | | 23:16 | AR23 | AR22 | AR21 | AR20 | AR19 | AR18 | AR17 | AR16 |
| | | 31:24 | AR31 | AR30 | AR29 | AR28 | AR27 | AR26 | AR25 | AR24 |
| 0xD4 | MCAN_TXBCR | 7:0 | CR7 | CR6 | CR5 | CR4 | CR3 | CR2 | CR1 | CR0 |
| | | 15:8 | CR15 | CR14 | CR13 | CR12 | CR11 | CR10 | CR9 | CR8 |
| | | 23:16 | CR23 | CR22 | CR21 | CR20 | CR19 | CR18 | CR17 | CR16 |
| | | 31:24 | CR31 | CR30 | CR29 | CR28 | CR27 | CR26 | CR25 | CR24 |
| 0xD8 | MCAN_TXBTO | 7:0 | TO7 | TO6 | TO5 | TO4 | TO3 | TO2 | TO1 | TO0 |
| | | 15:8 | TO15 | TO14 | TO13 | TO12 | TO11 | TO10 | TO9 | TO8 |
| | | 23:16 | TO23 | TO22 | TO21 | TO20 | TO19 | TO18 | TO17 | TO16 |
| | | 31:24 | TO31 | TO30 | TO29 | TO28 | TO27 | TO26 | TO25 | TO24 |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xDC | MCAN_TXBCF | 7:0 | CF7 | CF6 | CF5 | CF4 | CF3 | CF2 | CF1 | CF0 |
| | | 15:8 | CF15 | CF14 | CF13 | CF12 | CF11 | CF10 | CF9 | CF8 |
| | | 23:16 | CF23 | CF22 | CF21 | CF20 | CF19 | CF18 | CF17 | CF16 |
| | | 31:24 | CF31 | CF30 | CF29 | CF28 | CF27 | CF26 | CF25 | CF24 |
| 0xE0 | MCAN_TXBTIE | 7:0 | TIE7 | TIE6 | TIE5 | TIE4 | TIE3 | TIE2 | TIE1 | TIE0 |
| | | 15:8 | TIE15 | TIE14 | TIE13 | TIE12 | TIE11 | TIE10 | TIE9 | TIE8 |
| | | 23:16 | TIE23 | TIE22 | TIE21 | TIE20 | TIE19 | TIE18 | TIE17 | TIE16 |
| | | 31:24 | TIE31 | TIE30 | TIE29 | TIE28 | TIE27 | TIE26 | TIE25 | TIE24 |
| 0xE4 | MCAN_TXBCIE | 7:0 | CFIE7 | CFIE6 | CFIE5 | CFIE4 | CFIE3 | CFIE2 | CFIE1 | CFIE0 |
| | | 15:8 | CFIE15 | CFIE14 | CFIE13 | CFIE12 | CFIE11 | CFIE10 | CFIE9 | CFIE8 |
| | | 23:16 | CFIE23 | CFIE22 | CFIE21 | CFIE20 | CFIE19 | CFIE18 | CFIE17 | CFIE16 |
| | | 31:24 | CFIE31 | CFIE30 | CFIE29 | CFIE28 | CFIE27 | CFIE26 | CFIE25 | CFIE24 |
| 0xE8 ... 0xEF | Reserved | | | | | | | | | |
| 0xF0 | MCAN_TXEFC | 7:0 | EFSA[5:0] | | | | | | | |
| | | 15:8 | EFSA[13:6] | | | | | | | |
| | | 23:16 | | | EFS[5:0] | | | | | |
| | | 31:24 | | | EFWM[5:0] | | | | | |
| 0xF4 | MCAN_TXEFS | 7:0 | | | EFFL[5:0] | | | | | |
| | | 15:8 | | | | EFGI[4:0] | | | | |
| | | 23:16 | | | | EFPI[4:0] | | | | |
| | | 31:24 | | | | | | | TEFL | EFF |
| 0xF8 | MCAN_TXEFA | 7:0 | | | | EFAI[4:0] | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 52.6.1 MCAN Endian Register

**Name:** MCAN_ENDN
**Offset:** 0x04
**Reset:** 0x87654321
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ETV[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ETV[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ETV[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ETV[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |

**Bits 31:0 – ETV[31:0]** Endianness Test Value
The endianness test value is 0x87654321.

### 52.6.2 MCAN Customer Register

**Name:** MCAN_CUST
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CSV[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CSV[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CSV[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CSV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CSV[31:0]** Customer-specific Value
Customer-specific value.

### 52.6.3 MCAN Data Bit Timing and Prescaler Register

**Name:** MCAN_DBTP
**Offset:** 0x0C
**Reset:** 0x00000A33
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

The CAN bit time may be programmed in the range of 4 to 25 time quanta. The CAN time quantum may be programmed in the range of 1 to 32 CAN core clock periods. $t_q$ = (DBRP + 1) CAN core clock periods.

DTSEG1 is the sum of Prop_Seg and Phase_Seg1. DTSEG2 is Phase_Seg2.

Therefore the length of the bit time is (programmed values) [DTSEG1 + DTSEG2 + 3] $t_q$
or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] $t_q$.

The Information Processing Time (IPT) is zero, meaning the data for the next bit is available at the first clock edge after the sample point.

With a CAN core clock frequency of 8 MHz, the reset value of 0x00000A33 configures the MCAN for a fast bit rate of 500 kbit/s.

The bit rate configured for the CAN FD data phase via MCAN_DBTP must be higher than or equal to the bit rate configured for the arbitration phase via MCAN_NBTP.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TDC | | | DBRP[4:0] | | | | |
| Access | R/W | | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | DTSEG1[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 1 | 0 | 1 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DTSEG2[3:0] | | | | | DSJW[2:0] | | |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 1 | 1 | | 0 | 1 | 1 |

**Bit 23 – TDC** Transmitter Delay Compensation
0 (DISABLED): Transmitter Delay Compensation disabled.

1 (ENABLED): Transmitter Delay Compensation enabled.

**Bits 20:16 – DBRP[4:0]** Data Bit Rate Prescaler
The value by which the peripheral clock is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Bit Rate Prescaler are 0 to 31. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.

**Bits 12:8 – DTSEG1[4:0]** Data Time Segment Before Sample Point
0: Forbidden.

1 to 31: The duration of time segment is $t_q$ x (DTSEG1 + 1).

**Bits 7:4 – DTSEG2[3:0]** Data Time Segment After Sample Point
The duration of time segment is $t_q$ x (DTSEG2 + 1).

**Bits 2:0 – DSJW[2:0]** Data (Re) Synchronization Jump Width
The duration of a synchronization jump is $t_q$ x (DSJW + 1).

### 52.6.4 MCAN Test Register

**Name:** MCAN_TEST
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read/Write

Write access to the Test Register has to be enabled by setting bit MCAN_CCCR.TEST to ‘1’.

All MCAN Test Register functions are set to their reset values when bit MCAN_CCCR.TEST is cleared.

Loop Back mode and software control of pin CANTX are hardware test modes. Programming of TX ≠ 0 disturbs the message transfer on the CAN bus.

The reset value for MCAN_TEST.RX is undefined.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RX | TX[1:0] | | LBCK | | | | |
| Access | R | R/W | R/W | R/W | | | | |
| Reset | x | 0 | 0 | 0 | | | | |

**Bit 7 – RX** Receive Pin (read-only)
Monitors the actual value of pin CANRX.

The reset value for this bit is undefined.

| Value | Description |
|---|---|
| 0 | The CAN bus is dominant (CANRX = ‘0’). |
| 1 | The CAN bus is recessive (CANRX = ‘1’). |

**Bits 6:5 – TX[1:0]** Control of Transmit Pin (read/write)

| Value | Name | Description |
|---|---|---|
| 0 | RESET | Reset value, CANTX controlled by the CAN Core, updated at the end of the CAN bit time. |
| 1 | SAMPLE_POINT_MONITORING | Sample Point can be monitored at pin CANTX. |
| 2 | DOMINANT | Dominant (‘0’) level at pin CANTX. |

| Value | Name | Description |
|---|---|---|
| 3 | RECESSIVE | Recessive (‘1’) at pin CANTX. |

**Bit 4 – LBCK** Loop Back Mode (read/write)
0 (DISABLED): Reset value. Loop Back mode is disabled.

1 (ENABLED): Loop Back mode is enabled (see Test Modes).

### 52.6.5 MCAN RAM Watchdog Register

**Name:** MCAN_RWD
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** Read/Write

The RAM Watchdog monitors the Message RAM response time. A Message RAM access via the MCAN's Generic Master Interface starts the Message RAM Watchdog Counter with the value configured by MCAN_RWD.WDC. The counter is reloaded with MCAN_RWD.WDC when the Message RAM signals successful completion by activating its READY output. In case there is no response from the Message RAM until the counter has counted down to zero, the counter stops and interrupt flag MCAN_IR.WDI is set. The RAM Watchdog Counter is clocked by the system bus clock (peripheral clock).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WDV[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WDC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:8 – WDV[7:0]** Watchdog Value (read-only)
Watchdog Counter Value for the current message located in RAM.

**Bits 7:0 – WDC[7:0]** Watchdog Configuration (read/write)
Start value of the Message RAM Watchdog Counter. The counter is disabled when WDC is cleared.

### 52.6.6 MCAN CC Control Register

**Name:** MCAN_CCCR
**Offset:** 0x18
**Reset:** 0x00000001
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | TXP | EFBI | PXHD | | | BRSE | FDOE |
| Access | | R/W | R/W | R/W | | | R/W | R/W |
| Reset | | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TEST | DAR | MON | CSR | CSA | ASM | CCE | INIT |
| Access | R/W | R/W | R/W | R/W | R | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**Bit 14 – TXP** Transmit Pause (read/write, write protection)
If this bit is set, the MCAN pauses for two CAN bit times before starting the next transmission after itself has successfully transmitted a frame (see Tx Handling).

| Value | Description |
|---|---|
| 0 | Transmit pause disabled. |
| 1 | Transmit pause enabled. |

**Bit 13 – EFBI** Edge Filtering during Bus Integration (read/write, write protection)

| Value | Description |
|---|---|
| 0 | Edge filtering is disabled. |
| 1 | Edge filtering is enabled. Two consecutive dominant tq required to detect an edge for hard synchronization. |

**Bit 12 – PXHD** Protocol Exception Event Handling (read/write, write protection)

| Value | Description |
|---|---|
| 0 | Protocol exception handling enabled. |
| 1 | Protocol exception handling disabled. |

**Bit 9 – BRSE** Bit Rate Switching Enable (read/write, write protection)
0 (DISABLED): Bit rate switching for transmissions disabled.

1 (ENABLED): Bit rate switching for transmissions enabled.

**Bit 8 – FDOE** CAN FD Operation Enable (read/write, write protection)
0 (DISABLED): FD operation disabled.

1 (ENABLED): FD operation enabled.

**Bit 7 – TEST** Test Mode Enable (read/write, write protection against '1')
0 (DISABLED): Normal operation, MCAN_TEST register holds reset values.

1 (ENABLED): Test mode, write access to MCAN_TEST register enabled.

**Bit 6 – DAR** Disable Automatic Retransmission (read/write, write protection)
0 (AUTO_RETX): Automatic retransmission of messages not transmitted successfully enabled.

1 (NO_AUTO_RETX): Automatic retransmission disabled.

**Bit 5 – MON** Bus Monitoring Mode (read/write, write protection against '1')
0 (DISABLED): Bus Monitoring mode is disabled.

1 (ENABLED): Bus Monitoring mode is enabled.

**Bit 4 – CSR** Clock Stop Request (read/write)
0 (NO_CLOCK_STOP): No clock stop is requested.

1 (CLOCK_STOP): Clock stop requested. When clock stop is requested, first INIT and then CSA will be set after all pending transfer requests have been completed and the CAN bus reached idle.

**Bit 3 – CSA** Clock Stop Acknowledge (read-only)

| Value | Description |
|---|---|
| 0 | No clock stop acknowledged. |
| 1 | MCAN may be set in power down by stopping the peripheral clock and the CAN core clock. |

**Bit 2 – ASM** Restricted Operation Mode (read/write, write protection against '1')
For a description of the Restricted Operation mode see Restricted Operation Mode.

0 (NORMAL): Normal CAN operation.

1 (RESTRICTED): Restricted Operation mode active.

**Bit 1 – CCE** Configuration Change Enable (read/write, write protection)
0 (PROTECTED): The processor has no write access to the protected configuration registers.

1 (CONFIGURABLE): The processor has write access to the protected configuration registers (while MCAN_CCCR.INIT = '1').

**Bit 0 – INIT** Initialization (read/write)
Due to the synchronization mechanism between the two clock domains, there may be a delay until the value written to INIT can be read back. Therefore the programmer has to ensure that the previous value written to INIT has been accepted by reading INIT before setting INIT to a new value.

0 (DISABLED): Normal operation.

1 (ENABLED): Initialization is started.

### 52.6.7 MCAN Nominal Bit Timing and Prescaler Register

**Name:** MCAN_NBTP
**Offset:** 0x1C
**Reset:** 0x06000A03
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN_CCCR.

The CAN bit time may be programmed in the range of 4 to 385 time quanta. The CAN time quantum may be programmed in the range of 1 to 512 CAN core clock periods. $t_q = t_{core\ clock}$ x (NBRP + 1).

NTSEG1 is the sum of Prop_Seg and Phase_Seg1. NTSEG2 is Phase_Seg2.

Therefore the length of the bit time is (programmed values) [NTSEG1 + NTSEG2 + 3] $t_q$
or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] $t_q$.

The Information Processing Time (IPT) is zero, meaning the data for the next bit is available at the first clock edge after the sample point.

With a CAN core clock frequency of 8 MHz, the reset value of 0x06000A03 configures the MCAN for a bit rate of 500 kbit/s.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NSJW[6:0] | | | | | | | NBRP[8:8] |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NBRP[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NTSEG1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | NTSEG2[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

**Bits 31:25 – NSJW[6:0]** Nominal (Re) Synchronization Jump Width
0 to 127: The duration of a synchronization jump is $t_q$ x (NSJW + 1).

**Bits 24:16 – NBRP[8:0]** Nominal Bit Rate Prescaler
0 to 511: The value by which the oscillator frequency is divided for generating the CAN time quanta. The CAN time is built up from a multiple of this quanta. CAN time quantum (tq) = $t_{core\ clock}$ x (NBRP + 1)

**Bits 15:8 – NTSEG1[7:0]** Nominal Time Segment Before Sample Point

| Value | Description |
|---|---|
| 0 | Reserved; do not use. |
| 1 to 255 | The duration of time segment is $t_q$ x (NTSEG1 + 1). |

**Bits 6:0 – NTSEG2[6:0]** Nominal Time Segment After Sample Point

| Value | Description |
|---|---|
| 0 | Reserved; do not use. |
| 1 to 127 | The duration of time segment is $t_q$ x (NTSEG2 + 1). |

### 52.6.8 MCAN Timestamp Counter Configuration Register

**Name:** MCAN_TSCC
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read/Write

For a description of the Timestamp Counter see Timestamp Generation.

With CAN FD, an external counter is required for timestamp generation (TSS = 2).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TCP[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | TSS[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bits 19:16 – TCP[3:0]** Timestamp Counter Prescaler
Configures the timestamp and timeout counters time unit in multiples of CAN bit times [ 1…16 ]. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.

**Bits 1:0 – TSS[1:0]** Timestamp Select

| Value | Name | Description |
|---|---|---|
| 0 | ALWAYS_0 | Timestamp counter value always 0x0000 |
| 1 | TCP_INC | Timestamp counter value incremented according to TCP |
| 2 | EXT_TIMESTAMP | External timestamp counter value used |
| 3 | ALWAYS_0 | Timestamp counter value always 0x0000 |

### 52.6.9 MCAN Timestamp Counter Value Register

**Name:** MCAN_TSCV
**Offset:** 0x24
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TSC[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TSC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – TSC[15:0]** Timestamp Counter (cleared on write)
The internal/external Timestamp Counter value is captured on start of frame (both Receive and Transmit). When MCAN_TSCC.TSS = 1, the Timestamp Counter is incremented in multiples of CAN bit times [ 1… 16 ] depending on the configuration of MCAN_TSCC.TCP. A wrap around sets interrupt flag MCAN_IR.TSW. Write access resets the counter to zero.

When MCAN_TSCC.TSS = 2, TSC reflects the external Timestamp Counter value. Thus a write access has no impact.

**Note:** A “wrap around” is a change of the Timestamp Counter value from non-zero to zero not caused by write access to MCAN_TSCV.

### 52.6.10 MCAN Timeout Counter Configuration Register

**Name:** MCAN_TOCC
**Offset:** 0x28
**Reset:** 0xFFFF0000
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

For a description of the Timeout Counter, see Timeout Counter.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TOP[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TOP[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | TOS[1:0] | | ETOC |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 31:16 – TOP[15:0]** Timeout Period
Start value of the Timeout Counter (down-counter). Configures the Timeout Period.

**Bits 2:1 – TOS[1:0]** Timeout Select
When operating in Continuous mode, a write to MCAN_TOCV presets the counter to the value configured by MCAN_TOCC.TOP and continues down-counting. When the Timeout Counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by MCAN_TOCC.TOP. Down-counting is started when the first FIFO element is stored.

| Value | Name | Description |
|---|---|---|
| 0 | CONTINUOUS | Continuous operation. |
| 1 | TX_EV_TIMEOUT | Timeout controlled by Tx Event FIFO. |
| 2 | RX0_EV_TIMEOUT | Timeout controlled by Receive FIFO 0. |
| 3 | RX1_EV_TIMEOUT | Timeout controlled by Receive FIFO 1. |

**Bit 0 – ETOC** Enable Timeout Counter
0 (NO_TIMEOUT): Timeout Counter disabled.

1 (TOS_CONTROLLED): Timeout Counter enabled.

For use of timeout function with CAN FD, see Timeout Counter.

### 52.6.11 MCAN Timeout Counter Value Register

**Name:** MCAN_TOCV
**Offset:** 0x2C
**Reset:** 0x0000FFFF
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TOC[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TOC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 15:0 – TOC[15:0]** Timeout Counter (cleared on write)
The Timeout Counter is decremented in multiples of CAN bit times [ 1…16 ] depending on the configuration of MCAN_TSCC.TCP. When decremented to zero, interrupt flag MCAN_IR.TOO is set and the Timeout Counter is stopped. Start and reset/restart conditions are configured via MCAN_TOCC.TOS.

### 52.6.12 MCAN Error Counter Register

**Name:** MCAN_ECR
**Offset:** 0x40
**Reset:** 0x00000000
**Property:** Read-only

When MCAN_CCCR.ASM is set, the CAN protocol controller does not increment TEC and REC when a CAN protocol error is detected, but CEL is still incremented.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CEL[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RP | REC[6:0] | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TEC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – CEL[7:0]** CAN Error Logging (cleared on read)
The counter is incremented each time when a CAN protocol error causes the Transmit Error Counter or the Receive Error Counter to be incremented. It is reset by read access to CEL. The counter stops at 0xFF; the next increment of TEC or REC sets interrupt flag IR.ELO.

**Bit 15 – RP** Receive Error Passive

| Value | Description |
|---|---|
| 0 | The Receive Error Counter is below the error passive level of 128. |
| 1 | The Receive Error Counter has reached the error passive level of 128. |

**Bits 14:8 – REC[6:0]** Receive Error Counter
Actual state of the Receive Error Counter, values between 0 and 127.

**Bits 7:0 – TEC[7:0]** Transmit Error Counter
Actual state of the Transmit Error Counter, values between 0 and 255.

### 52.6.13 MCAN Protocol Status Register

**Name:** MCAN_PSR
**Offset:** 0x44
**Reset:** 0x00000707
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | TDCV[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | PXE | RFDF | RBRS | RESI | DLEC[2:0] | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BO | EW | EP | ACT[1:0] | | LEC[2:0] | | |
| Access | R | R | R | R | R | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

**Bits 22:16 – TDCV[6:0]** Transmitter Delay Compensation Value
0 to 127: Position of the secondary sample point, in CAN core clock periods, defined by the sum of the measured delay from CANTX to CANRX and MCAN_TDCR.TDCO.

**Bit 14 – PXE** Protocol Exception Event (cleared on read)

| Value | Description |
|---|---|
| 0 | No protocol exception event occurred since last read access |
| 1 | Protocol exception event occurred |

**Bit 13 – RFDF** Received a CAN FD Message (cleared on read)
This bit is set independently from acceptance filtering.

| Value | Description |
|---|---|
| 0 | Since this bit was reset by the CPU, no CAN FD message has been received |
| 1 | Message in CAN FD format with FDF flag set has been received |

**Bit 12 – RBRS** BRS Flag of Last Received CAN FD Message (cleared on read)
This bit is set together with RFDF, independently from acceptance filtering.

| Value | Description |
|---|---|
| 0 | Last received CAN FD message did not have its BRS flag set. |

| Value | Description |
|---|---|
| 1 | Last received CAN FD message had its BRS flag set. |

**Bit 11 – RESI** ESI Flag of Last Received CAN FD Message (cleared on read)
This bit is set together with RFDF, independently from acceptance filtering.

| Value | Description |
|---|---|
| 0 | Last received CAN FD message did not have its ESI flag set. |
| 1 | Last received CAN FD message had its ESI flag set. |

**Bits 10:8 – DLEC[2:0]** Data Phase Last Error Code (set to 111 on read)
Type of last error that occurred in the data phase of a CAN FD format frame with its BRS flag set. Coding is the same as for LEC. This field will be cleared to zero when a CAN FD format frame with its BRS flag set has been transferred (reception or transmission) without error.

**Bit 7 – BO** Bus_Off Status

| Value | Description |
|---|---|
| 0 | The MCAN is not Bus_Off. |
| 1 | The MCAN is in Bus_Off state. |

**Bit 6 – EW** Warning Status

| Value | Description |
|---|---|
| 0 | Both error counters are below the Error_Warning limit of 96. |
| 1 | At least one of error counter has reached the Error_Warning limit of 96. |

**Bit 5 – EP** Error Passive

| Value | Description |
|---|---|
| 0 | The MCAN is in the Error_Active state. It normally takes part in bus communication and sends an active error flag when an error has been detected. |
| 1 | The MCAN is in the Error_Passive state. |

**Bits 4:3 – ACT[1:0]** Activity
Monitors the CAN communication state of the CAN module.

| Value | Name | Description |
|---|---|---|
| 0 | SYNCHRONIZING | Node is synchronizing on CAN communication |
| 1 | IDLE | Node is neither receiver nor transmitter |
| 2 | RECEIVER | Node is operating as receiver |
| 3 | TRANSMITTER | Node is operating as transmitter |

**Bits 2:0 – LEC[2:0]** Last Error Code (set to 111 on read)
The LEC indicates the type of the last error to occur on the CAN bus. This field is cleared when a message has been transferred (reception or transmission) without error.

| Value | Name | Description |
|---|---|---|
| 0 | NO_ERROR | No error occurred since LEC has been reset by successful reception or transmission. |
| 1 | STUFF_ERROR | More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed. |

| Value | Name | Description |
|---|---|---|
| 2 | FORM_ERROR | A fixed format part of a received frame has the wrong format. |
| 3 | ACK_ERROR | The message transmitted by the MCAN was not acknowledged by another node. |
| 4 | BIT1_ERROR | During transmission of a message (with the exception of the arbitration field), the device tried to send a recessive level (bit of logical value ‘1’), but the monitored bus value was dominant. |
| 5 | BIT0_ERROR | During transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device tried to send a dominant level (data or identifier bit logical value ‘0’), but the monitored bus value was recessive. During Bus_Off recovery, this status is set each time a sequence of 11 recessive bits has been monitored. This enables the processor to monitor the proceeding of the Bus_Off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed). |
| 6 | CRC_ERROR | The CRC check sum of a received message was incorrect. The CRC of an incoming message does not match the CRC calculated from the received data. |
| 7 | NO_CHANGE | Any read access to the Protocol Status Register re-initializes the LEC to ‘7’. When the LEC shows value ‘7’, no CAN bus event was detected since the last processor read access to the Protocol Status Register. |

### 52.6.14 MCAN Transmitter Delay Compensation Register

**Name:** MCAN_TDCR
**Offset:** 0x48
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | TDCO[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TDCF[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 14:8 – TDCO[6:0]** Transmitter Delay Compensation Offset
0 to 127: Offset value, in CAN core clock periods, defining the distance between the measured delay from CANTX to CANRX and the secondary sample point.

**Bits 6:0 – TDCF[6:0]** Transmitter Delay Compensation Filter
0 to 127: defines the minimum value for the SSP position, in CAN core clock periods. Dominant edges on CANRX that would result in an earlier SSP position are ignored for transmitter delay measurement. The feature is enabled when TDCF is configured to a value greater than TDCO.

### 52.6.15 MCAN Interrupt Register

**Name:** MCAN_IR
**Offset:** 0x50
**Reset:** 0x00000000
**Property:** Read/Write

The flags are set when one of the listed conditions is detected (edge-sensitive). The flags remain set until the processor clears them. A flag is cleared by writing a '1' to the corresponding bit position. Writing a '0' has no effect. A hard reset will clear the register. The configuration of IE controls whether an interrupt is generated. The configuration of ILS controls on which interrupt line an interrupt is signalled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | ARA | PED | PEA | WDI | BO | EW |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | EP | ELO | | | DRX | TOO | MRAF | TSW |
| Access | R/W | R/W | | | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TEFL | TEFF | TEFW | TEFN | TFE | TCF | TC | HPM |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RF1L | RF1F | RF1W | RF1N | RF0L | RF0F | RF0W | RF0N |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 29 – ARA** Access to Reserved Address

| Value | Description |
|---|---|
| 0 | No access to reserved address occurred |
| 1 | Access to reserved address occurred |

**Bit 28 – PED** Protocol Error in Data Phase

| Value | Description |
|---|---|
| 0 | No protocol error in data phase |
| 1 | Protocol error in data phase detected (MCAN_PSR.DLEC differs from 0 or 7) |

**Bit 27 – PEA** Protocol Error in Arbitration Phase

| Value | Description |
|---|---|
| 0 | No protocol error in arbitration phase |
| 1 | Protocol error in arbitration phase detected (MCAN_PSR.LEC differs from 0 or 7) |

**Bit 26 – WDI** Watchdog Interrupt

| Value | Description |
|---|---|
| 0 | No Message RAM Watchdog event occurred. |
| 1 | Message RAM Watchdog event due to missing READY. |

**Bit 25 – BO** Bus_Off Status

| Value | Description |
|---|---|
| 0 | Bus_Off status unchanged. |
| 1 | Bus_Off status changed. |

**Bit 24 – EW** Warning Status

| Value | Description |
|---|---|
| 0 | Error_Warning status unchanged. |
| 1 | Error_Warning status changed. |

**Bit 23 – EP** Error Passive

| Value | Description |
|---|---|
| 0 | Error_Passive status unchanged. |
| 1 | Error_Passive status changed. |

**Bit 22 – ELO** Error Logging Overflow

| Value | Description |
|---|---|
| 0 | CAN Error Logging Counter did not overflow. |
| 1 | Overflow of CAN Error Logging Counter occurred. |

**Bit 19 – DRX** Message stored to Dedicated Receive Buffer
The flag is set whenever a received message has been stored into a dedicated Receive Buffer.

| Value | Description |
|---|---|
| 0 | No Receive Buffer updated. |
| 1 | At least one received message stored into a Receive Buffer. |

**Bit 18 – TOO** Timeout Occurred

| Value | Description |
|---|---|
| 0 | No timeout. |
| 1 | Timeout reached. |

**Bit 17 – MRAF** Message RAM Access Failure
The flag is set, when the Rx Handler

• has not completed acceptance filtering or storage of an accepted message until the arbitration field of the following message has been received. In this case acceptance filtering or message storage is aborted and the Rx Handler starts processing of the following message.

• was not able to write a message to the Message RAM. In this case message storage is aborted.

In both cases the FIFO put index is not updated resp. the New Data flag for a dedicated Receive Buffer is not set, a partly stored message is overwritten when the next message is stored to this location.

The flag is also set when the Tx Handler was not able to read a message from the Message RAM in time. In this case message transmission is aborted. In case of a Tx Handler access failure the MCAN is switched into Restricted Operation mode (see Restricted Operation Mode). To leave Restricted Operation mode, the processor has to reset MCAN_CCCR.ASM.

| Value | Description |
|---|---|
| 0 | No Message RAM access failure occurred. |
| 1 | Message RAM access failure occurred. |

**Bit 16 – TSW** Timestamp Wraparound

| Value | Description |
|---|---|
| 0 | No timestamp counter wrap-around. |
| 1 | Timestamp counter wrapped around. |

**Bit 15 – TEFL** Tx Event FIFO Element Lost

| Value | Description |
|---|---|
| 0 | No Tx Event FIFO element lost. |
| 1 | Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero. |

**Bit 14 – TEFF** Tx Event FIFO Full

| Value | Description |
|---|---|
| 0 | Tx Event FIFO not full. |
| 1 | Tx Event FIFO full. |

**Bit 13 – TEFW** Tx Event FIFO Watermark Reached

| Value | Description |
|---|---|
| 0 | Tx Event FIFO fill level below watermark. |
| 1 | Tx Event FIFO fill level reached watermark. |

**Bit 12 – TEFN** Tx Event FIFO New Entry

| Value | Description |
|---|---|
| 0 | Tx Event FIFO unchanged. |
| 1 | Tx Handler wrote Tx Event FIFO element. |

**Bit 11 – TFE** Tx FIFO Empty

| Value | Description |
|---|---|
| 0 | Tx FIFO non-empty. |
| 1 | Tx FIFO empty. |

**Bit 10 – TCF** Transmission Cancellation Finished

| Value | Description |
|---|---|
| 0 | No transmission cancellation finished. |
| 1 | Transmission cancellation finished. |

**Bit 9 – TC** Transmission Completed

| Value | Description |
|---|---|
| 0 | No transmission completed. |
| 1 | Transmission completed. |

**Bit 8 – HPM** High Priority Message

| Value | Description |
|---|---|
| 0 | No high priority message received. |
| 1 | High priority message received. |

**Bit 7 – RF1L** Receive FIFO 1 Message Lost

| Value | Description |
|---|---|
| 0 | No Receive FIFO 1 message lost. |
| 1 | Receive FIFO 1 message lost, also set after write attempt to Receive FIFO 1 of size zero. |

**Bit 6 – RF1F** Receive FIFO 1 Full

| Value | Description |
|---|---|
| 0 | Receive FIFO 1 not full. |
| 1 | Receive FIFO 1 full. |

**Bit 5 – RF1W** Receive FIFO 1 Watermark Reached

| Value | Description |
|---|---|
| 0 | Receive FIFO 1 fill level below watermark. |
| 1 | Receive FIFO 1 fill level reached watermark. |

**Bit 4 – RF1N** Receive FIFO 1 New Message

| Value | Description |
|---|---|
| 0 | No new message written to Receive FIFO 1. |
| 1 | New message written to Receive FIFO 1. |

**Bit 3 – RF0L** Receive FIFO 0 Message Lost

| Value | Description |
|---|---|
| 0 | No Receive FIFO 0 message lost. |
| 1 | Receive FIFO 0 message lost, also set after write attempt to Receive FIFO 0 of size zero. |

**Bit 2 – RF0F** Receive FIFO 0 Full

| Value | Description |
|---|---|
| 0 | Receive FIFO 0 not full. |
| 1 | Receive FIFO 0 full. |

**Bit 1 – RF0W** Receive FIFO 0 Watermark Reached

| Value | Description |
|---|---|
| 0 | Receive FIFO 0 fill level below watermark. |
| 1 | Receive FIFO 0 fill level reached watermark. |

**Bit 0 – RF0N** Receive FIFO 0 New Message

| Value | Description |
|---|---|
| 0 | No new message written to Receive FIFO 0. |
| 1 | New message written to Receive FIFO 0. |

### 52.6.16 MCAN Interrupt Enable Register

**Name:** MCAN_IE
**Offset:** 0x54
**Reset:** 0x00000000
**Property:** Read/Write

The following configuration values are valid for all listed bit names of this register:

0: Disables the corresponding interrupt.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | ARAE | PEDE | PEAE | WDIE | BOE | EWE |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | EPE | ELOE | | | DRXE | TOOE | MRAFE | TSWE |
| Access | R/W | R/W | | | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TEFLE | TEFFE | TEFWE | TEFNE | TFEE | TCFE | TCE | HPME |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RF1LE | RF1FE | RF1WE | RF1NE | RF0LE | RF0FE | RF0WE | RF0NE |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 29 – ARAE** Access to Reserved Address Enable

**Bit 28 – PEDE** Protocol Error in Data Phase Enable

**Bit 27 – PEAE** Protocol Error in Arbitration Phase Enable

**Bit 26 – WDIE** Watchdog Interrupt Enable

**Bit 25 – BOE** Bus_Off Status Interrupt Enable

**Bit 24 – EWE** Warning Status Interrupt Enable

**Bit 23 – EPE** Error Passive Interrupt Enable

**Bit 22 – ELOE** Error Logging Overflow Interrupt Enable

**Bit 19 – DRXE** Message stored to Dedicated Receive Buffer Interrupt Enable

**Bit 18 – TOOE** Timeout Occurred Interrupt Enable

**Bit 17 – MRAFE** Message RAM Access Failure Interrupt Enable

**Bit 16 – TSWE** Timestamp Wraparound Interrupt Enable

**Bit 15 – TEFLE** Tx Event FIFO Event Lost Interrupt Enable

**Bit 14 – TEFFE** Tx Event FIFO Full Interrupt Enable

**Bit 13 – TEFWE** Tx Event FIFO Watermark Reached Interrupt Enable

**Bit 12 – TEFNE** Tx Event FIFO New Entry Interrupt Enable

**Bit 11 – TFEE** Tx FIFO Empty Interrupt Enable

**Bit 10 – TCFE** Transmission Cancellation Finished Interrupt Enable

**Bit 9 – TCE** Transmission Completed Interrupt Enable

**Bit 8 – HPME** High Priority Message Interrupt Enable

**Bit 7 – RF1LE** Receive FIFO 1 Message Lost Interrupt Enable

**Bit 6 – RF1FE** Receive FIFO 1 Full Interrupt Enable

**Bit 5 – RF1WE** Receive FIFO 1 Watermark Reached Interrupt Enable

**Bit 4 – RF1NE** Receive FIFO 1 New Message Interrupt Enable

**Bit 3 – RF0LE** Receive FIFO 0 Message Lost Interrupt Enable

**Bit 2 – RF0FE** Receive FIFO 0 Full Interrupt Enable

**Bit 1 – RF0WE** Receive FIFO 0 Watermark Reached Interrupt Enable

**Bit 0 – RF0NE** Receive FIFO 0 New Message Interrupt Enable

### 52.6.17 MCAN Interrupt Line Select Register

**Name:** MCAN_ILS
**Offset:** 0x58
**Reset:** 0x00000000
**Property:** Read/Write

The Interrupt Line Select register assigns an interrupt generated by a specific interrupt flag from the Interrupt Register to one of the two module interrupt lines.

0: Interrupt assigned to interrupt line MCAN_INT0.

1: Interrupt assigned to interrupt line MCAN_INT1.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | ARAL | PEDL | PEAL | WDIL | BOL | EWL |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | EPL | ELOL | | | DRXL | TOOL | MRAFL | TSWL |
| Access | R/W | R/W | | | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TEFLL | TEFFL | TEFWL | TEFNL | TFEL | TCFL | TCL | HPML |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RF1LL | RF1FL | RF1WL | RF1NL | RF0LL | RF0FL | RF0WL | RF0NL |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 29 – ARAL** Access to Reserved Address Line

**Bit 28 – PEDL** Protocol Error in Data Phase Line

**Bit 27 – PEAL** Protocol Error in Arbitration Phase Line

**Bit 26 – WDIL** Watchdog Interrupt Line

**Bit 25 – BOL** Bus_Off Status Interrupt Line

**Bit 24 – EWL** Warning Status Interrupt Line

**Bit 23 – EPL** Error Passive Interrupt Line

**Bit 22 – ELOL** Error Logging Overflow Interrupt Line

**Bit 19 – DRXL** Message stored to Dedicated Receive Buffer Interrupt Line

**Bit 18 – TOOL** Timeout Occurred Interrupt Line

**Bit 17 – MRAFL** Message RAM Access Failure Interrupt Line

**Bit 16 – TSWL** Timestamp Wraparound Interrupt Line

**Bit 15 – TEFLL** Tx Event FIFO Event Lost Interrupt Line

**Bit 14 – TEFFL** Tx Event FIFO Full Interrupt Line

**Bit 13 – TEFWL** Tx Event FIFO Watermark Reached Interrupt Line

**Bit 12 – TEFNL** Tx Event FIFO New Entry Interrupt Line

**Bit 11 – TFEL** Tx FIFO Empty Interrupt Line

**Bit 10 – TCFL** Transmission Cancellation Finished Interrupt Line

**Bit 9 – TCL** Transmission Completed Interrupt Line

**Bit 8 – HPML** High Priority Message Interrupt Line

**Bit 7 – RF1LL** Receive FIFO 1 Message Lost Interrupt Line

**Bit 6 – RF1FL** Receive FIFO 1 Full Interrupt Line

**Bit 5 – RF1WL** Receive FIFO 1 Watermark Reached Interrupt Line

**Bit 4 – RF1NL** Receive FIFO 1 New Message Interrupt Line

**Bit 3 – RF0LL** Receive FIFO 0 Message Lost Interrupt Line

**Bit 2 – RF0FL** Receive FIFO 0 Full Interrupt Line

**Bit 1 – RF0WL** Receive FIFO 0 Watermark Reached Interrupt Line

**Bit 0 – RF0NL** Receive FIFO 0 New Message Interrupt Line

### 52.6.18 MCAN Interrupt Line Enable

**Name:** MCAN_ILE
**Offset:** 0x5C
**Reset:** 0x00000000
**Property:** Read/Write

Each of the two interrupt lines to the processor can be enabled/disabled separately by programming bits EINT0 and EINT1.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | EINT1 | EINT0 |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – EINT1** Enable Interrupt Line 1

| Value | Description |
|---|---|
| 0 | Interrupt line MCAN_INT1 disabled. |
| 1 | Interrupt line MCAN_INT1 enabled. |

**Bit 0 – EINT0** Enable Interrupt Line 0

| Value | Description |
|---|---|
| 0 | Interrupt line MCAN_INT0 disabled. |
| 1 | Interrupt line MCAN_INT0 enabled. |

### 52.6.19 MCAN Global Filter Configuration

**Name:** MCAN_GFC
**Offset:** 0x80
**Reset:** 0x00000000
**Property:** Read/Write

Global settings for Message ID filtering. The Global Filter Configuration controls the filter path for standard and extended messages as illustrated in Standard Message ID Filter Path and Extended Message ID Filter Path.

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | ANFS[1:0] | | ANFE[1:0] | | RRFS | RRFE |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:4 – ANFS[1:0]** Accept Non-matching Frames Standard
Defines how received messages with 11-bit IDs that do not match any element of the filter list are treated.

| Value | Name | Description |
|---|---|---|
| 0 | RX_FIFO_0 | Accept in Rx FIFO 0 |
| 1 | RX_FIFO_1 | Accept in Rx FIFO 1 |
| 2-3 | REJECTED | Message rejected |

**Bits 3:2 – ANFE[1:0]** Accept Non-matching Frames Extended
Defines how received messages with 29-bit IDs that do not match any element of the filter list are treated.

| Value | Name | Description |
|---|---|---|
| 0 | RX_FIFO_0 | Accept in Rx FIFO 0 |
| 1 | RX_FIFO_1 | Accept in Rx FIFO 1 |
| 2-3 | REJECTED | Message rejected |

**Bit 1 – RRFS** Reject Remote Frames Standard
0 (FILTER): Filter remote frames with 11-bit standard IDs.

1 (REJECT): Reject all remote frames with 11-bit standard IDs.

**Bit 0 – RRFE** Reject Remote Frames Extended
0 (FILTER): Filter remote frames with 29-bit extended IDs.

1 (REJECT): Reject all remote frames with 29-bit extended IDs.

### 52.6.20 MCAN Standard ID Filter Configuration

**Name:** MCAN_SIDFC
**Offset:** 0x84
**Reset:** 0x00000000
**Property:** Read/Write

Settings for 11-bit standard Message ID filtering. The Standard ID Filter Configuration controls the filter path for standard messages as illustrated in Standard Message ID Filter Path.

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | LSS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FLSSA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FLSSA[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 23:16 – LSS[7:0]** List Size Standard
>128: Values greater than 128 are interpreted as 128.

| Value | Description |
|---|---|
| 0 | No standard Message ID filter. |
| 1-128 | Number of standard Message ID filter elements. |

**Bits 15:2 – FLSSA[13:0]** Filter List Standard Start Address
Start address of standard Message ID filter list (32-bit word address, see Message RAM Configuration).

Write FLSSA with the bits [15:2] of the 32-bit address.

### 52.6.21 MCAN Extended ID Filter Configuration

**Name:** MCAN_XIDFC
**Offset:** 0x88
**Reset:** 0x00000000
**Property:** Read/Write

Settings for 29-bit extended Message ID filtering. The Extended ID Filter Configuration controls the filter path for standard messages as described in Extended Message ID Filter Path.

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | LSE[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FLESA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FLESA[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 22:16 – LSE[6:0]** List Size Extended

| Value | Description |
|---|---|
| 0 | No extended Message ID filter. |
| 1-64 | Number of extended Message ID filter elements. |
| >64 | Values greater than 64 are interpreted as 64. |

**Bits 15:2 – FLESA[13:0]** Filter List Extended Start Address
Start address of extended Message ID filter list (32-bit word address, see Message RAM Configuration).

Write FLESA with the bits [15:2] of the 32-bit address.

### 52.6.22 MCAN Extended ID AND Mask

**Name:** MCAN_XIDAM
**Offset:** 0x90
**Reset:** 0x1FFFFFFF
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | EIDM[28:24] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 1 | 1 | 1 | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | EIDM[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EIDM[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EIDM[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 28:0 – EIDM[28:0]** Extended ID Mask
For acceptance filtering of extended frames the Extended ID AND Mask is ANDed with the Message ID of a received frame. Intended for masking of 29-bit IDs in SAE J1939. With the reset value of all bits set to one the mask is not active.

### 52.6.23 MCAN High Priority Message Status

**Name:** MCAN_HPMS
**Offset:** 0x94
**Reset:** 0x00000000
**Property:** Read-only

This register is updated every time a Message ID filter element configured to generate a priority event matches. This can be used to monitor the status of incoming high priority messages and to enable fast access to these messages.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FLST | FIDX[6:0] | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MSI[1:0] | | BIDX[5:0] | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 15 – FLST** Filter List
Indicates the filter list of the matching filter element.

| Value | Description |
|---|---|
| 0 | Standard filter list |
| 1 | Extended filter list |

**Bits 14:8 – FIDX[6:0]** Filter Index
Index of matching filter element. Range is 0 to MCAN_SIDFC.LSS - 1 resp. MCAN_XIDFC.LSE - 1.

**Bits 7:6 – MSI[1:0]** Message Storage Indicator

| Value | Name | Description |
|---|---|---|
| 0 | NO_FIFO_SEL | No FIFO selected. |
| 1 | LOST | FIFO message lost. |
| 2 | FIFO_0 | Message stored in FIFO 0. |
| 3 | FIFO_1 | Message stored in FIFO 1. |

**Bits 5:0 – BIDX[5:0]** Buffer Index
Index of Receive FIFO element to which the message was stored. Only valid when MSI[1] = ‘1’.

### 52.6.24 MCAN New Data 1

**Name:** MCAN_NDAT1
**Offset:** 0x98
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ND31 | ND30 | ND29 | ND28 | ND27 | ND26 | ND25 | ND24 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ND23 | ND22 | ND21 | ND20 | ND19 | ND18 | ND17 | ND16 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ND15 | ND14 | ND13 | ND12 | ND11 | ND10 | ND9 | ND8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ND7 | ND6 | ND5 | ND4 | ND3 | ND2 | ND1 | ND0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – NDx** New Data
The register holds the New Data flags of Receive Buffers 0 to 31. The flags are set when the respective Receive Buffer has been updated from a received frame. The flags remain set until the processor clears them. A flag is cleared by writing a ‘1’ to the corresponding bit position. Writing a ‘0’ has no effect. A hard reset will clear the register.

| Value | Description |
|---|---|
| 0 | Receive Buffer not updated |
| 1 | Receive Buffer updated from new message |

### 52.6.25 MCAN New Data 2

**Name:** MCAN_NDAT2
**Offset:** 0x9C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ND63 | ND62 | ND61 | ND60 | ND59 | ND58 | ND57 | ND56 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ND55 | ND54 | ND53 | ND52 | ND51 | ND50 | ND49 | ND48 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ND47 | ND46 | ND45 | ND44 | ND43 | ND42 | ND41 | ND40 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ND39 | ND38 | ND37 | ND36 | ND35 | ND34 | ND33 | ND32 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – NDx** New Data
The register holds the New Data flags of Receive Buffers 32 to 63. The flags are set when the respective Receive Buffer has been updated from a received frame. The flags remain set until the processor clears them. A flag is cleared by writing a ‘1’ to the corresponding bit position. Writing a ‘0’ has no effect. A hard reset will clear the register.

| Value | Description |
|---|---|
| 0 | Receive Buffer not updated. |
| 1 | Receive Buffer updated from new message. |

### 52.6.26 MCAN Receive FIFO 0 Configuration

**Name:** MCAN_RXF0C
**Offset:** 0xA0
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | F0OM | F0WM[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | F0S[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | F0SA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | F0SA[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bit 31 – F0OM** FIFO 0 Operation Mode
FIFO 0 can be operated in Blocking or in Overwrite mode (see Rx FIFOs).

| Value | Description |
|---|---|
| 0 | FIFO 0 Blocking mode. |
| 1 | FIFO 0 Overwrite mode. |

**Bits 30:24 – F0WM[6:0]** Receive FIFO 0 Watermark

| Value | Description |
|---|---|
| 0 | Watermark interrupt disabled. |
| 1-64 | Level for Receive FIFO 0 watermark interrupt (MCAN_IR.RF0W). |
| >64 | Watermark interrupt disabled. |

**Bits 22:16 – F0S[6:0]** Receive FIFO 0 Size
The Receive FIFO 0 elements are indexed from 0 to F0S-1.

| Value | Description |
|---|---|
| 0 | No Receive FIFO 0 |
| 1-64 | Number of Receive FIFO 0 elements. |
| >64 | Values greater than 64 are interpreted as 64. |

**Bits 15:2 – F0SA[13:0]** Receive FIFO 0 Start Address
Start address of Receive FIFO 0 in Message RAM (32-bit word address, see Message RAM Configuration).

Write F0SA with the bits [15:2] of the 32-bit address.

### 52.6.27 MCAN Receive FIFO 0 Status

**Name:** MCAN_RXF0S
**Offset:** 0xA4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RF0L | F0F |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | F0PI[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | F0GI[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | F0FL[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 25 – RF0L** Receive FIFO 0 Message Lost
This bit is a copy of interrupt flag MCAN_IR.RF0L. When MCAN_IR.RF0L is reset, this bit is also reset.

Overwriting the oldest message when MCAN_RXF0C.F0OM = ‘1’ will not set this flag.

| Value | Description |
|---|---|
| 0 | No Receive FIFO 0 message lost |
| 1 | Receive FIFO 0 message lost, also set after write attempt to Receive FIFO 0 of size zero |

**Bit 24 – F0F** Receive FIFO 0 Full

| Value | Description |
|---|---|
| 0 | Receive FIFO 0 not full. |
| 1 | Receive FIFO 0 full. |

**Bits 21:16 – F0PI[5:0]** Receive FIFO 0 Put Index
Receive FIFO 0 write index pointer, range 0 to 63.

**Bits 13:8 – F0GI[5:0]** Receive FIFO 0 Get Index
Receive FIFO 0 read index pointer, range 0 to 63.

**Bits 6:0 – F0FL[6:0]** Receive FIFO 0 Fill Level
Number of elements stored in Receive FIFO 0, range 0 to 64.

### 52.6.28 MCAN Receive FIFO 0 Acknowledge

**Name:** MCAN_RXF0A
**Offset:** 0xA8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | F0AI[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:0 – F0AI[5:0]** Receive FIFO 0 Acknowledge Index
After the processor has read a message or a sequence of messages from Receive FIFO 0 it has to write the buffer index of the last element read from Receive FIFO 0 to F0AI. This will set the Receive FIFO 0 Get Index MCAN_RXF0S.F0GI to F0AI + 1 and update the FIFO 0 Fill Level MCAN_RXF0S.F0FL.

### 52.6.29 MCAN Receive Buffer Configuration

**Name:** MCAN_RXBC
**Offset:** 0xAC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RBSA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RBSA[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 15:2 – RBSA[13:0]** Receive Buffer Start Address
Configures the start address of the Receive Buffers section in the Message RAM (32-bit word address, see Message RAM Configuration). Also used to reference debug messages A,B,C.

Write RBSA with the bits [15:2] of the 32-bit address.

### 52.6.30 MCAN Receive FIFO 1 Configuration

**Name:** MCAN_RXF1C
**Offset:** 0xB0
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | F1OM | F1WM[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | F1S[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | F1SA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | F1SA[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bit 31 – F1OM** FIFO 1 Operation Mode
FIFO 1 can be operated in Blocking or in Overwrite mode (see Rx FIFOs).

| Value | Description |
|---|---|
| 0 | FIFO 1 Blocking mode. |
| 1 | FIFO 1 Overwrite mode. |

**Bits 30:24 – F1WM[6:0]** Receive FIFO 1 Watermark

| Value | Description |
|---|---|
| 0 | Watermark interrupt disabled |
| 1-64 | Level for Receive FIFO 1 watermark interrupt (MCAN_IR.RF1W). |
| >64 | Watermark interrupt disabled. |

**Bits 22:16 – F1S[6:0]** Receive FIFO 1 Size
The elements in Receive FIFO 1 are indexed from 0 to F1S - 1.

| Value | Description |
|---|---|
| 0 | No Receive FIFO 1 |
| 1-64 | Number of elements in Receive FIFO 1. |
| >64 | Values greater than 64 are interpreted as 64. |

**Bits 15:2 – F1SA[13:0]** Receive FIFO 1 Start Address
Start address of Receive FIFO 1 in Message RAM (32-bit word address, see Message RAM Configuration).

Write F1SA with the bits [15:2] of the 32-bit address.

### 52.6.31 MCAN Receive FIFO 1 Status

**Name:** MCAN_RXF1S
**Offset:** 0xB4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DMS[1:0] | | | | | | RF1L | F1F |
| Access | R | R | | | | | R | R |
| Reset | 0 | 0 | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | F1PI[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | F1GI[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | F1FL[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:30 – DMS[1:0]** Debug Message Status

| Value | Name | Description |
|---|---|---|
| 0 | IDLE | Idle state, wait for reception of debug messages, DMA request is cleared. |
| 1 | MSG_A | Debug message A received. |
| 2 | MSG_AB | Debug messages A, B received. |
| 3 | MSG_ABC | Debug messages A, B, C received, DMA request is set. |

**Bit 25 – RF1L** Receive FIFO 1 Message Lost
This bit is a copy of interrupt flag IR.RF1L. When IR.RF1L is reset, this bit is also reset.

Overwriting the oldest message when MCAN_RXF1C.F1OM = ‘1’ will not set this flag.

| Value | Description |
|---|---|
| 0 | No Receive FIFO 1 message lost. |
| 1 | Receive FIFO 1 message lost, also set after write attempt to Receive FIFO 1 of size zero. |

**Bit 24 – F1F** Receive FIFO 1 Full

| Value | Description |
|---|---|
| 0 | Receive FIFO 1 not full. |
| 1 | Receive FIFO 1 full. |

**Bits 21:16 – F1PI[5:0]** Receive FIFO 1 Put Index
Receive FIFO 1 write index pointer, range 0 to 63.

**Bits 13:8 – F1GI[5:0]** Receive FIFO 1 Get Index
Receive FIFO 1 read index pointer, range 0 to 63.

**Bits 6:0 – F1FL[6:0]** Receive FIFO 1 Fill Level
Number of elements stored in Receive FIFO 1, range 0 to 64.

### 52.6.32 MCAN Receive FIFO 1 Acknowledge

**Name:** MCAN_RXF1A
**Offset:** 0xB8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | F1AI[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:0 – F1AI[5:0]** Receive FIFO 1 Acknowledge Index
After the processor has read a message or a sequence of messages from Receive FIFO 1 it has to write the buffer index of the last element read from Receive FIFO 1 to F1AI. This will set the Receive FIFO 1 Get Index MCAN_RXF1S.F1GI to F1AI + 1 and update the FIFO 1 Fill Level MCAN_RXF1S.F1FL.

### 52.6.33 MCAN Receive Buffer / FIFO Element Size Configuration

**Name:** MCAN_RXESC
**Offset:** 0xBC
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

Configures the number of data bytes belonging to a Receive Buffer / Receive FIFO element. Data field sizes >8 bytes are intended for CAN FD operation only.

In case the data field size of an accepted CAN frame exceeds the data field size configured for the matching Receive Buffer or Receive FIFO, only the number of bytes as configured by MCAN_RXESC are stored to the Receive Buffer resp. Receive FIFO element. The rest of the frame's data field is ignored.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | RBDS[2:0] | | |
| Reset | | | | | | R/W | R/W | R/W |
| | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | F1DS[2:0] | | | | F0DS[2:0] | | |
| Reset | | R/W | R/W | R/W | | R/W | R/W | R/W |
| | | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bits 10:8 – RBDS[2:0]** Receive Buffer Data Field Size

| Value | Name | Description |
|---|---|---|
| 0 | 8_BYTE | 8-byte data field |
| 1 | 12_BYTE | 12-byte data field |
| 2 | 16_BYTE | 16-byte data field |
| 3 | 20_BYTE | 20-byte data field |
| 4 | 24_BYTE | 24-byte data field |
| 5 | 32_BYTE | 32-byte data field |
| 6 | 48_BYTE | 48-byte data field |
| 7 | 64_BYTE | 64-byte data field |

**Bits 6:4 – F1DS[2:0]** Receive FIFO 1 Data Field Size

| Value | Name | Description |
|---|---|---|
| 0 | 8_BYTE | 8-byte data field |
| 1 | 12_BYTE | 12-byte data field |
| 2 | 16_BYTE | 16-byte data field |
| 3 | 20_BYTE | 20-byte data field |
| 4 | 24_BYTE | 24-byte data field |
| 5 | 32_BYTE | 32-byte data field |
| 6 | 48_BYTE | 48-byte data field |
| 7 | 64_BYTE | 64-byte data field |

**Bits 2:0 – F0DS[2:0]** Receive FIFO 0 Data Field Size

| Value | Name | Description |
|---|---|---|
| 0 | 8_BYTE | 8-byte data field |
| 1 | 12_BYTE | 12-byte data field |
| 2 | 16_BYTE | 16-byte data field |
| 3 | 20_BYTE | 20-byte data field |
| 4 | 24_BYTE | 24-byte data field |
| 5 | 32_BYTE | 32-byte data field |
| 6 | 48_BYTE | 48-byte data field |
| 7 | 64_BYTE | 64-byte data field |

### 52.6.34 MCAN Tx Buffer Configuration

**Name:** MCAN_TXBC
**Offset:** 0xC0
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

The sum of TFQS and NDTB may not exceed 32. There is no check for erroneous configurations. The Tx Buffers section in the Message RAM starts with the dedicated Tx Buffers.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | TFQM | TFQS[5:0] | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | NDTB[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TBSA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TBSA[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bit 30 – TFQM** Tx FIFO/Queue Mode

| Value | Description |
|---|---|
| 0 | Tx FIFO operation. |
| 1 | Tx Queue operation. |

**Bits 29:24 – TFQS[5:0]** Transmit FIFO/Queue Size

| Value | Description |
|---|---|
| 0 | No Tx FIFO/Queue. |
| 1-32 | Number of Tx Buffers used for Tx FIFO/Queue. |
| >32 | Values greater than 32 are interpreted as 32. |

**Bits 21:16 – NDTB[5:0]** Number of Dedicated Transmit Buffers

| Value | Description |
|---|---|
| 0 | No dedicated Tx Buffers. |
| 1-32 | Number of dedicated Tx Buffers. |
| >32 | Values greater than 32 are interpreted as 32. |

**Bits 15:2 – TBSA[13:0]** Tx Buffers Start Address
Start address of Tx Buffers section in Message RAM (32-bit word address, see Message RAM Configuration).

Write TBSA with the bits [15:2] of the 32-bit address.

### 52.6.35 MCAN Tx FIFO/Queue Status

**Name:** MCAN_TXFQS
**Offset:** 0xC4
**Reset:** 0x00000000
**Property:** Read-only

The Tx FIFO/Queue status is related to the pending Tx requests listed in register MCAN_TXBRP. Therefore the effect of Add/Cancellation requests may be delayed due to a running Tx scan (MCAN_TXBRP not yet updated).

In case of mixed configurations where dedicated Tx Buffers are combined with a Tx FIFO or a Tx Queue, the Put and Get Indices indicate the number of the Tx Buffer starting with the first dedicated Tx Buffers. Example: For a configuration of 12 dedicated Tx Buffers and a Tx FIFO of 20 Buffers a Put Index of 15 points to the fourth buffer of the Tx FIFO.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | TFQF | TFQPI[4:0] | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | TFGI[4:0] | | | | |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TFFL[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 21 – TFQF** Tx FIFO/Queue Full

| Value | Description |
|---|---|
| 0 | Tx FIFO/Queue not full. |
| 1 | Tx FIFO/Queue full. |

**Bits 20:16 – TFQPI[4:0]** Tx FIFO/Queue Put Index
Tx FIFO/Queue write index pointer, range 0 to 31.

**Bits 12:8 – TFGI[4:0]** Tx FIFO Get Index
Tx FIFO read index pointer, range 0 to 31. Read as zero when Tx Queue operation is configured (MCAN_TXBC.TFQM = ‘1’).

**Bits 5:0 – TFFL[5:0]** Tx FIFO Free Level
Number of consecutive free Tx FIFO elements starting from TFGI, range 0 to 32. Read as zero when Tx Queue operation is configured (MCAN_TXBC.TFQM = ‘1’).

### 52.6.36 MCAN Tx Buffer Element Size Configuration

**Name:** MCAN_TXESC
**Offset:** 0xC8
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

Configures the number of data bytes belonging to a Tx Buffer element. Data field sizes > 8 bytes are intended for CAN FD operation only.

In case the data length code DLC of a Tx Buffer element is configured to a value higher than the Tx Buffer data field size MCAN_TXESC.TBDS, the bytes not defined by the Tx Buffer are transmitted as “0xCC” (padding bytes).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | TBDS[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 2:0 – TBDS[2:0]** Tx Buffer Data Field Size

| Value | Name | Description |
|---|---|---|
| 0 | 8_BYTE | 8-byte data field |
| 1 | 12_BYTE | 12-byte data field |
| 2 | 16_BYTE | 16-byte data field |
| 3 | 20_BYTE | 20-byte data field |
| 4 | 24_BYTE | 24-byte data field |
| 5 | 32_BYTE | 32-byte data field |
| 6 | 48_BYTE | 48- byte data field |
| 7 | 64_BYTE | 64-byte data field |

### 52.6.37 MCAN Transmit Buffer Request Pending

**Name:** MCAN_TXBRP
**Offset:** 0xCC
**Reset:** 0x00000000
**Property:** Read-only

MCAN_TXBRP bits which are set while a Tx scan is in progress are not considered during this particular Tx scan. In case a cancellation is requested for such a Tx Buffer, this Add Request is cancelled immediately, the corresponding MCAN_TXBRP bit is reset.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TRP31 | TRP30 | TRP29 | TRP28 | TRP27 | TRP26 | TRP25 | TRP24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TRP23 | TRP22 | TRP21 | TRP20 | TRP19 | TRP18 | TRP17 | TRP16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TRP15 | TRP14 | TRP13 | TRP12 | TRP11 | TRP10 | TRP9 | TRP8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TRP7 | TRP6 | TRP5 | TRP4 | TRP3 | TRP2 | TRP1 | TRP0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – TRPx** Transmission Request Pending for Buffer x
Each Tx Buffer has its own Transmission Request Pending bit. The bits are set via register MCAN_TXBAR. The bits are reset after a requested transmission has completed or has been cancelled via register MCAN_TXBCR.

TXBRP bits are set only for those Tx Buffers configured via MCAN_TXBC. After a MCAN_TXBRP bit has been set, a Tx scan (see Tx Handling) is started to check for the pending Tx request with the highest priority (Tx Buffer with lowest Message ID).

A cancellation request resets the corresponding transmission request pending bit of register MCAN_TXBRP. In case a transmission has already been started when a cancellation is requested, this is done at the end of the transmission, regardless whether the transmission was successful or not. The cancellation request bits are reset directly after the corresponding TXBRP bit has been reset.

After a cancellation has been requested, a finished cancellation is signalled via MCAN_TXBCF.

- after successful transmission together with the corresponding MCAN_TXBTO bit.
- when the transmission has not yet been started at the point of cancellation.

• when the transmission has been aborted due to lost arbitration.

• when an error occurred during frame transmission.

In DAR mode, all transmissions are automatically cancelled if they are not successful. The corresponding MCAN_TXBCF bit is set for all unsuccessful transmissions.

| Value | Description |
|---|---|
| 0 | No transmission request pending |
| 1 | Transmission request pending |

### 52.6.38 MCAN Transmit Buffer Add Request

**Name:** MCAN_TXBAR
**Offset:** 0xD0
**Reset:** 0x00000000
**Property:** Read/Write

If an add request is applied for a Transmit Buffer with pending transmission request (corresponding MCAN_TXBRP bit already set), this Add Request is ignored.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | AR31 | AR30 | AR29 | AR28 | AR27 | AR26 | AR25 | AR24 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | AR23 | AR22 | AR21 | AR20 | AR19 | AR18 | AR17 | AR16 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | AR15 | AR14 | AR13 | AR12 | AR11 | AR10 | AR9 | AR8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | AR7 | AR6 | AR5 | AR4 | AR3 | AR2 | AR1 | AR0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – ARx** Add Request for Transmit Buffer x
Each Transmit Buffer has its own Add Request bit. Writing a ‘1’ will set the corresponding Add Request bit; writing a ‘0’ has no impact. This enables the processor to set transmission requests for multiple Transmit Buffers with one write to MCAN_TXBAR. MCAN_TXBAR bits are set only for those Transmit Buffers configured via TXBC. When no Transmit scan is running, the bits are reset immediately, else the bits remain set until the Transmit scan process has completed.

| Value | Description |
|---|---|
| 0 | No transmission request added. |
| 1 | Transmission requested added. |

### 52.6.39 MCAN Transmit Buffer Cancellation Request

**Name:** MCAN_TXBCR
**Offset:** 0xD4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CR31 | CR30 | CR29 | CR28 | CR27 | CR26 | CR25 | CR24 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CR23 | CR22 | CR21 | CR20 | CR19 | CR18 | CR17 | CR16 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CR15 | CR14 | CR13 | CR12 | CR11 | CR10 | CR9 | CR8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CR7 | CR6 | CR5 | CR4 | CR3 | CR2 | CR1 | CR0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – CRx** Cancellation Request for Transmit Buffer x
Each Transmit Buffer has its own Cancellation Request bit. Writing a ‘1’ will set the corresponding Cancellation Request bit; writing a ‘0’ has no impact. This enables the processor to set cancellation requests for multiple Transmit Buffers with one write to MCAN_TXBCR. MCAN_TXBCR bits are set only for those Transmit Buffers configured via TXBC. The bits remain set until the corresponding bit of MCAN_TXBRP is reset.

| Value | Description |
|---|---|
| 0 | No cancellation pending. |
| 1 | Cancellation pending. |

### 52.6.40 MCAN Transmit Buffer Transmission Occurred

**Name:** MCAN_TXBTO
**Offset:** 0xD8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TO31 | TO30 | TO29 | TO28 | TO27 | TO26 | TO25 | TO24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TO23 | TO22 | TO21 | TO20 | TO19 | TO18 | TO17 | TO16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TO15 | TO14 | TO13 | TO12 | TO11 | TO10 | TO9 | TO8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TO7 | TO6 | TO5 | TO4 | TO3 | TO2 | TO1 | TO0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – TOx** Transmission Occurred for Buffer x
Each Transmit Buffer has its own Transmission Occurred bit. The bits are set when the corresponding MCAN_TXBRP bit is cleared after a successful transmission. The bits are reset when a new transmission is requested by writing a ‘1’ to the corresponding bit of register MCAN_TXBAR.

| Value | Description |
|---|---|
| 0 | No transmission occurred. |
| 1 | Transmission occurred. |

### 52.6.41 MCAN Transmit Buffer Cancellation Finished

**Name:** MCAN_TXBCF
**Offset:** 0xDC
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CF31 | CF30 | CF29 | CF28 | CF27 | CF26 | CF25 | CF24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CF23 | CF22 | CF21 | CF20 | CF19 | CF18 | CF17 | CF16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CF15 | CF14 | CF13 | CF12 | CF11 | CF10 | CF9 | CF8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CF7 | CF6 | CF5 | CF4 | CF3 | CF2 | CF1 | CF0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – CFx** Cancellation Finished for Transmit Buffer x
Each Transmit Buffer has its own Cancellation Finished bit. The bits are set when the corresponding MCAN_TXBRP bit is cleared after a cancellation was requested via MCAN_TXBCR. In case the corresponding MCAN_TXBRP bit was not set at the point of cancellation, CF is set immediately. The bits are reset when a new transmission is requested by writing a ‘1’ to the corresponding bit of register MCAN_TXBAR.

| Value | Description |
|---|---|
| 0 | No transmit buffer cancellation. |
| 1 | Transmit buffer cancellation finished. |

### 52.6.42 MCAN Transmit Buffer Transmission Interrupt Enable

**Name:** MCAN_TXBTIE
**Offset:** 0xE0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TIE31 | TIE30 | TIE29 | TIE28 | TIE27 | TIE26 | TIE25 | TIE24 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TIE23 | TIE22 | TIE21 | TIE20 | TIE19 | TIE18 | TIE17 | TIE16 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TIE15 | TIE14 | TIE13 | TIE12 | TIE11 | TIE10 | TIE9 | TIE8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TIE7 | TIE6 | TIE5 | TIE4 | TIE3 | TIE2 | TIE1 | TIE0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – TIEx** Transmission Interrupt Enable for Buffer x
Each Transmit Buffer has its own Transmission Interrupt Enable bit.

| Value | Description |
|---|---|
| 0 | Transmission interrupt disabled |
| 1 | Transmission interrupt enable |

### 52.6.43 MCAN Transmit Buffer Cancellation Finished Interrupt Enable

**Name:** MCAN_TXBCIE
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CFIE31 | CFIE30 | CFIE29 | CFIE28 | CFIE27 | CFIE26 | CFIE25 | CFIE24 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CFIE23 | CFIE22 | CFIE21 | CFIE20 | CFIE19 | CFIE18 | CFIE17 | CFIE16 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CFIE15 | CFIE14 | CFIE13 | CFIE12 | CFIE11 | CFIE10 | CFIE9 | CFIE8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CFIE7 | CFIE6 | CFIE5 | CFIE4 | CFIE3 | CFIE2 | CFIE1 | CFIE0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – CFIEx** Cancellation Finished Interrupt Enable for Transmit Buffer x
Each Transmit Buffer has its own Cancellation Finished Interrupt Enable bit.

| Value | Description |
|---|---|
| 0 | Cancellation finished interrupt disabled. |
| 1 | Cancellation finished interrupt enabled. |

### 52.6.44 MCAN Transmit Event FIFO Configuration

**Name:** MCAN_TXEFC
**Offset:** 0xF0
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | EFWM[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | EFS[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EFSA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EFSA[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 29:24 – EFWM[5:0]** Event FIFO Watermark

| Value | Description |
|---|---|
| 0 | Watermark interrupt disabled. |
| 1-32 | Level for Tx Event FIFO watermark interrupt (MCAN_IR.TEFW). |
| >32 | Watermark interrupt disabled. |

**Bits 21:16 – EFS[5:0]** Event FIFO Size
The Tx Event FIFO elements are indexed from 0 to EFS - 1.

| Value | Description |
|---|---|
| 0 | Tx Event FIFO disabled. |
| 1-32 | Number of Tx Event FIFO elements. |
| >32 | Values greater than 32 are interpreted as 32. |

**Bits 15:2 – EFSA[13:0]** Event FIFO Start Address
Start address of Tx Event FIFO in Message RAM (32-bit word address, see Message RAM Configuration).

Write EFSA with the bits [15:2] of the 32-bit address.

### 52.6.45 MCAN Tx Event FIFO Status

**Name:** MCAN_TXEFS
**Offset:** 0xF4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | TEFL | EFF |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | EFPI[4:0] | | | | |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | EFGI[4:0] | | | | |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | EFFL[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 25 – TEFL** Tx Event FIFO Element Lost
This bit is a copy of interrupt flag MCAN_IR.TEFL. When MCAN_IR.TEFL is reset, this bit is also reset.

| Value | Description |
|---|---|
| 0 | No Tx Event FIFO element lost. |
| 1 | Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero. |

**Bit 24 – EFF** Event FIFO Full

| Value | Description |
|---|---|
| 0 | Tx Event FIFO not full. |
| 1 | Tx Event FIFO full. |

**Bits 20:16 – EFPI[4:0]** Event FIFO Put Index
Tx Event FIFO write index pointer, range 0 to 31.

**Bits 12:8 – EFGI[4:0]** Event FIFO Get Index
Tx Event FIFO read index pointer, range 0 to 31.

**Bits 5:0 – EFFL[5:0]** Event FIFO Fill Level
Number of elements stored in Tx Event FIFO, range 0 to 32.

### 52.6.46 MCAN Tx Event FIFO Acknowledge

**Name:** MCAN_TXEFA
**Offset:** 0xF8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | EFAI[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 4:0 – EFAI[4:0]** Event FIFO Acknowledge Index
After the processor has read an element or a sequence of elements from the Tx Event FIFO, it has to write the index of the last element read from Tx Event FIFO to EFAI. This will set the Tx Event FIFO Get Index MCAN_TXEFS.EFGI to EFAI + 1 and update the FIFO 0 Fill Level MCAN_TXEFS.EFFL.

# 53. Timer Counter (TC)

## 53.1 Description

A Timer Counter (TC) module includes three identical TC channels. The number of implemented TC modules is device-specific.

Each TC channel can be independently programmed to perform a wide range of functions including frequency measurement, event counting, interval measurement, pulse generation, delay timing and pulse width modulation.

Each channel has three external clock inputs, five internal clock inputs and two multipurpose input/output signals which can be configured by the user. Each channel drives an internal interrupt signal which can be programmed to generate processor interrupts.

The TC embeds a quadrature decoder (QDEC) connected in front of the timers and driven by TIOA0, TIOB0 and TIOB1 inputs. When enabled, the QDEC performs the input lines filtering, decoding of quadrature signals and connects to the timers/counters in order to read the position and speed of the motor through the user interface.

The TC block has two global registers which act upon all TC channels:

- Block Control register (TC_BCR)—allows channels to be started simultaneously with the same instruction
- Block Mode register (TC_BMR)—defines the external clock inputs for each channel, allowing them to be chained

## 53.2 Embedded Characteristics

- Total of 6 Channels
- 32-bit Channel Size
- Wide Range of Functions Including:
  - Frequency measurement
  - Event counting
  - Interval measurement
  - Pulse generation
  - Delay timing
  - Pulse Width Modulation
  - Up/down capabilities
  - Quadrature decoder
  - 2-bit Gray up/down count for stepper motor
- Each Channel is User-Configurable and Contains:
  - Three external clock inputs
  - Five Internal clock inputs
  - Two multipurpose input/output signals acting as trigger event
  - Trigger/capture events can be directly synchronized by PWM signals
- Internal Interrupt Signal

- Read of the Capture Registers by the DMAC
- Compare Event Fault Generation for PWM
- Register Write Protection

## 53.3 Block Diagram

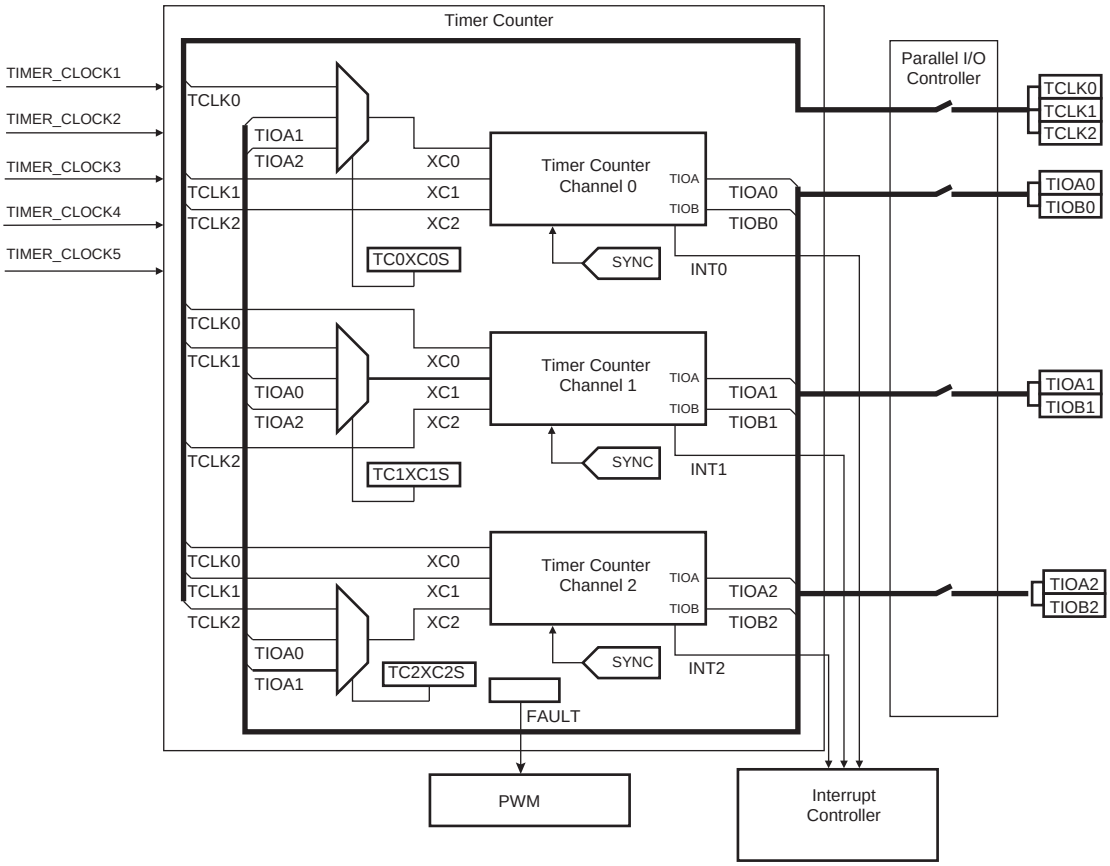
**Table 53-1. Timer Counter Clock Assignment**

| Name | Definition |
|---|---|
| TIMER_CLOCK1 | GCLK[35], GCLK[36] |
| TIMER_CLOCK2 | System bus clock divided by 8 |
| TIMER_CLOCK3 | System bus clock divided by 32 |
| TIMER_CLOCK4 | System bus clock divided by 128 |
| TIMER_CLOCK5 (See Notes) | slow_clock |

**Note:**

1. The GCLK frequency must be at least three times lower than peripheral clock frequency.

**Figure 53-1. Timer Counter Block Diagram**




**Note:**
The QDEC connections are detailed in Figure 53-17.

**Table 53-2. Channel Signal Description**

| Signal Name | Description |
|---|---|
| XC0, XC1, XC2 | External Clock Inputs |
| TIOAx | Capture Mode: Timer Counter Input<br>Waveform Mode: Timer Counter Output |
| TIOBx | Capture Mode: Timer Counter Input<br>Waveform Mode: Timer Counter Input/Output |
| INT | Interrupt Signal Output (internal signal) |
| SYNC | Synchronization Input Signal (from configuration register) |

## 53.4 Pin List

**Table 53-3. Pin List**

| Pin Name | Description | Type |
|---|---|---|
| TCLK0–TCLK2 | External Clock Input | Input |
| TIOA0–TIOA2 | I/O Line A | I/O |
| TIOB0–TIOB2 | I/O Line B | I/O |

## 53.5 Product Dependencies

### 53.5.1 I/O Lines

The pins used for interfacing the compliant external devices may be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the TC pins to their peripheral functions.

### 53.5.2 Power Management

The TC is clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the Timer Counter clock.

### 53.5.3 Interrupt Sources

The TC has an interrupt line connected to the interrupt controller. Handling the TC interrupt requires programming the interrupt controller before configuring the TC.

### 53.5.4 Synchronization Inputs from PWM

The TC has trigger/capture inputs internally connected to the PWM. Refer to “Synchronization with PWM” and to the implementation of the Pulse Width Modulation (PWM) in this product.

### 53.5.5 Fault Output

The TC has the FAULT output internally connected to the fault input of PWM. Refer to “Fault Mode” and to the implementation of the Pulse Width Modulation (PWM) in this product.

## 53.6 Functional Description

### 53.6.1 Description

All channels of the Timer Counter are independent and identical in operation except when the QDEC is enabled. The registers for channel programming are listed in 53.7 Register Summary.

### 53.6.2 32-bit Counter

Each 32-bit channel is organized around a 32-bit counter. The value of the counter is incremented at each positive edge of the selected clock. When the counter has reached the value $2^{32}$-1 and passes to zero, an overflow occurs and the COVFS bit in the Interrupt Status register (TC_SR) is set.

The current value of the counter is accessible in real time by reading the Counter Value register (TC_CV). The counter can be reset by a trigger. In this case, the counter value passes to zero on the next valid edge of the selected clock.

### 53.6.3 Clock Selection

At block level, input clock signals of each channel can be connected either to the external inputs TCLKx, or to the internal I/O signals TIOAx for chaining(1) by programming the Block Mode register (TC_BMR). See Clock Chaining Selection.

Each channel can independently select an internal or external clock source for its counter(2):

- External clock signals: XC0, XC1 or XC2
- Internal clock signals: GCLK[35], GCLK[36], System bus clock divided by 8, System bus clock divided by 32, System bus clock divided by 128, slow_clock

This selection is made by the TCCLKS bits in the Channel Mode register (TC_CMRx).

The selected clock can be inverted with TC_CMRx.CLKI. This allows counting on the opposite edges of the clock.

The burst function allows the clock to be validated when an external signal is high. The BURST parameter in the TC_CMRx defines this signal (none, XC0, XC1, XC2). See Clock Selection.

**Note:**

1. In Waveform mode, to chain two timers, it is mandatory to initialize some parameters:
   – Configure TIOx outputs to 1 or 0 by writing the required value to TC_CMRx.ASWTRG.
   – Bit TC_BCR.SYNC must be written to 1 to start the channels at the same time.
2. In all cases, if an external clock or asynchronous internal clock GCLK is used, the duration of each of its levels must be longer than the peripheral clock period, so the clock frequency will be at least 2.5 times lower than the peripheral clock.

**Figure 53-2. Clock Chaining Selection**

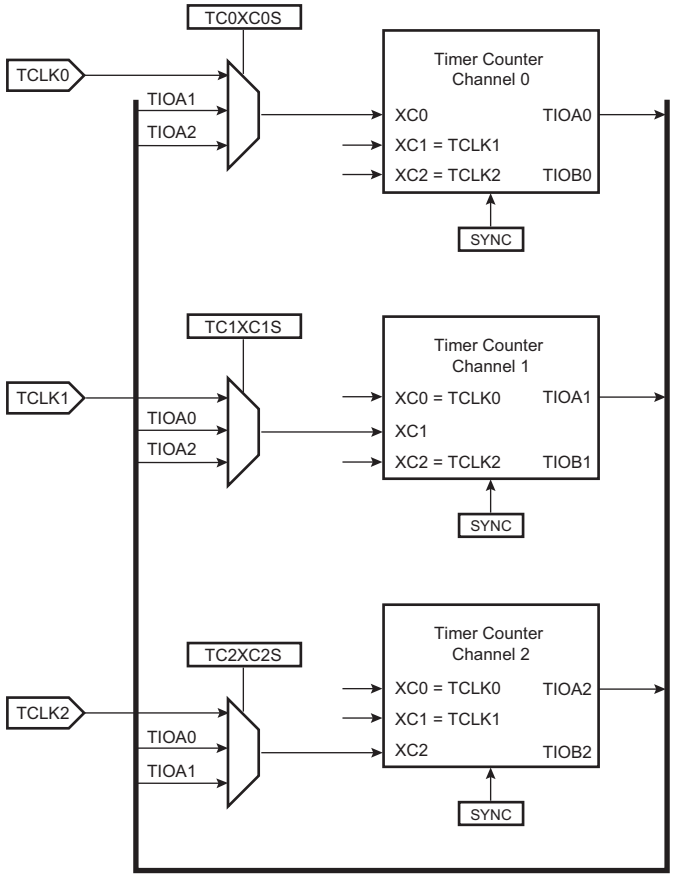
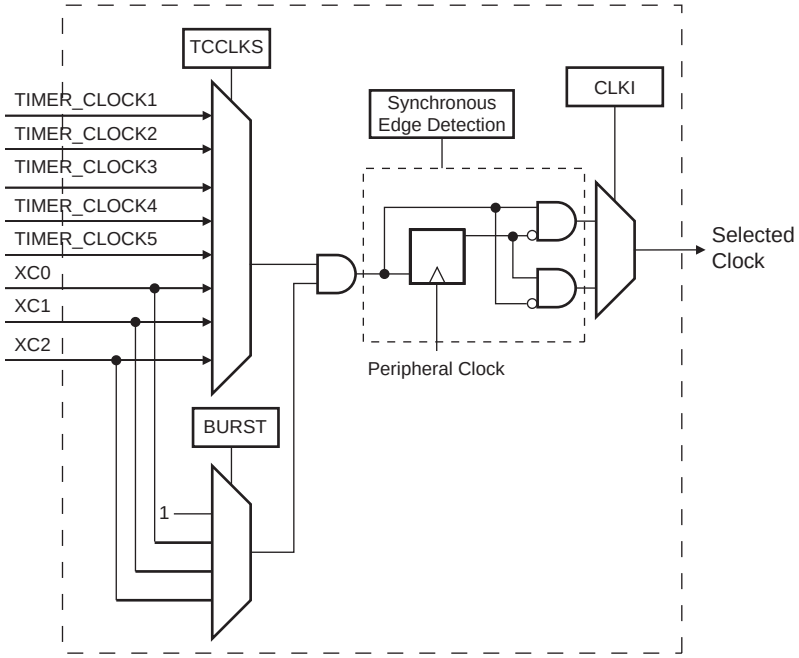



**Figure 53-3. Clock Selection**

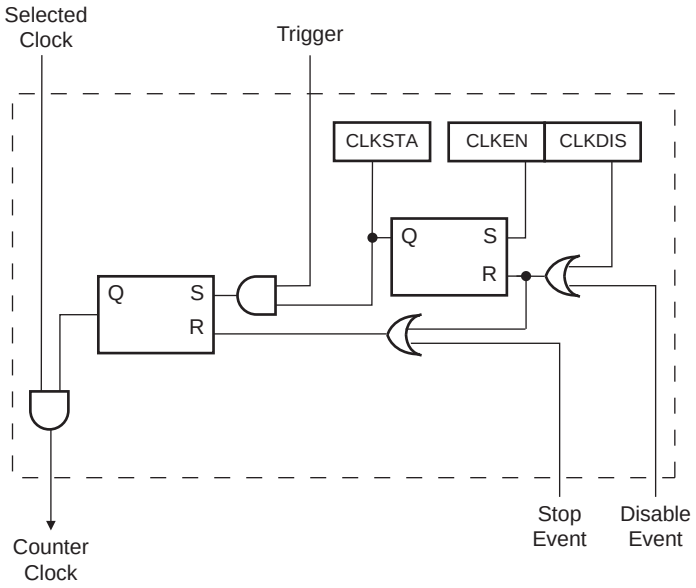


### 53.6.4 Clock Control

The clock of each counter can be controlled in two different ways: it can be enabled/disabled and started/stopped, as shown in the following figure.

- The clock can be enabled or disabled by the user with the CLKEN and the CLKDIS commands in the Channel Control register (TC_CCR). In Capture mode it can be disabled by an RB load event if TC_CMRx.LDBDIS is set to ‘1’. In Waveform mode, it can be disabled by an RC Compare event if TC_CMRx.CPCDIS is set to ‘1’. When disabled, the start or the stop actions have no effect: only a CLKEN command in the TC_CCR can reenable the clock. When the clock is enabled, TC_SR.CLKSTA is set.
- The clock can also be started or stopped: a trigger (software, synchro, external or compare) always starts the clock. The clock can be stopped by an RB load event in Capture mode (TC_CMRx.LDBSTOP = 1) or an RC compare event in Waveform mode (TC_CMRx.CPCSTOP = 1). The start and the stop commands are effective only if the clock is enabled.

**Figure 53-4. Clock Control**




### 53.6.5 Operating Modes

Each channel can operate independently in two different modes:

- Capture mode provides measurement on signals.
- Waveform mode provides wave generation.

The TC operating mode is programmed with TC_CMRx.WAVE.

In Capture mode, TIOAx and TIOBx are configured as inputs.

In Waveform mode, TIOAx is always configured to be an output and TIOBx is an output if it is not selected to be the external trigger.

### 53.6.6 Trigger

A trigger resets the counter and starts the counter clock. Three types of triggers are common to both modes, and a fourth external trigger is available to each mode.

Regardless of the trigger used, it will be taken into account at the following active edge of the selected clock. This means that the counter value can be read differently from zero just after a trigger, especially when a low frequency signal is selected as the clock.

The following triggers are common to both modes:

- Software Trigger: Each channel has a software trigger, available by setting TC_CCR.SWTRG.
- SYNC: Each channel has a synchronization signal SYNC. When asserted, this signal has the same effect as a software trigger. The SYNC signals of all channels are asserted simultaneously by writing TC_BCR with SYNC set.
- Compare RC Trigger: RC is implemented in each channel and can provide a trigger when the counter value matches the RC value if TC_CMRx.CPCTRG is set .

The channel can also be configured to have an external trigger. In Capture mode, the external trigger signal can be selected between TIOAx and TIOBx. In Waveform mode, an external event can be programmed on one of the following signals: TIOBx, XC0, XC1 or XC2. This external event can then be programmed to perform a trigger by setting TC_CMRx.ENETRG.

If an external trigger is used, the duration of the pulses must be longer than the peripheral clock period in order to be detected.

### 53.6.7 Capture Mode

Capture mode is entered by clearing TC_CMRx.WAVE.

Capture mode allows the TC channel to perform measurements such as pulse timing, frequency, period, duty cycle and phase on TIOAx and TIOBx signals which are considered as inputs.

Figure 53-6 shows the configuration of the TC channel when programmed in Capture mode.

### 53.6.8 Capture Registers A and B

Registers A and B (TC_RA and TC_RB) are used as capture registers. They can be loaded with the counter value when a programmable event occurs on the signal TIOAx.

TC_CMRx.LDRA defines the TIOAx selected edge for the loading of TC_RA, and TC_CMRx.LDRB defines the TIOAx selected edge for the loading of TC_RB.

The subsampling ratio defined by TC_CMRx.SBSMPLR is applied to these selected edges, so that the loading of Register A and Register B occurs once every 1, 2, 4, 8 or 16 selected edges.

TC_RA is loaded only if it has not been loaded since the last trigger or if TC_RB has been loaded since the last loading of TC_RA.

TC_RB is loaded only if TC_RA has been loaded since the last trigger or the last loading of TC_RB.

Loading TC_RA or TC_RB before the read of the last value loaded sets TC_SR.LOVRS. In this case, the old value is overwritten.

When DMA is used, the Register AB (TC_RAB) address must be configured as source address of the transfer. TC_RAB provides the next unread value from TC_RA and TC_RB. It may be read by the DMA after a request has been triggered upon loading TC_RA or TC_RB.

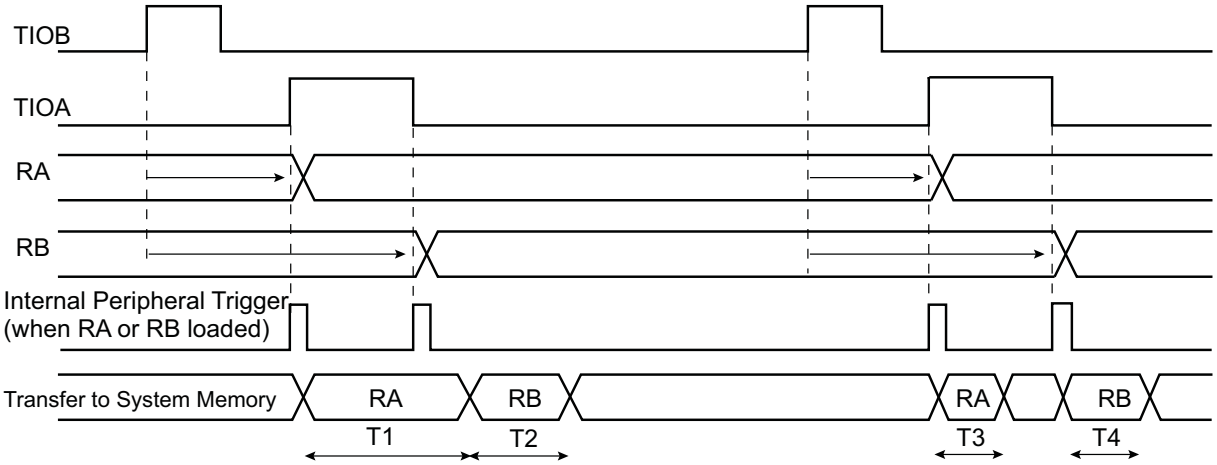
### 53.6.9 Transfer with DMAC in Capture Mode

The DMAC can perform access from the TC to system memory in Capture mode only.

The following figure illustrates how TC_RA and TC_RB can be loaded in the system memory without processor intervention.

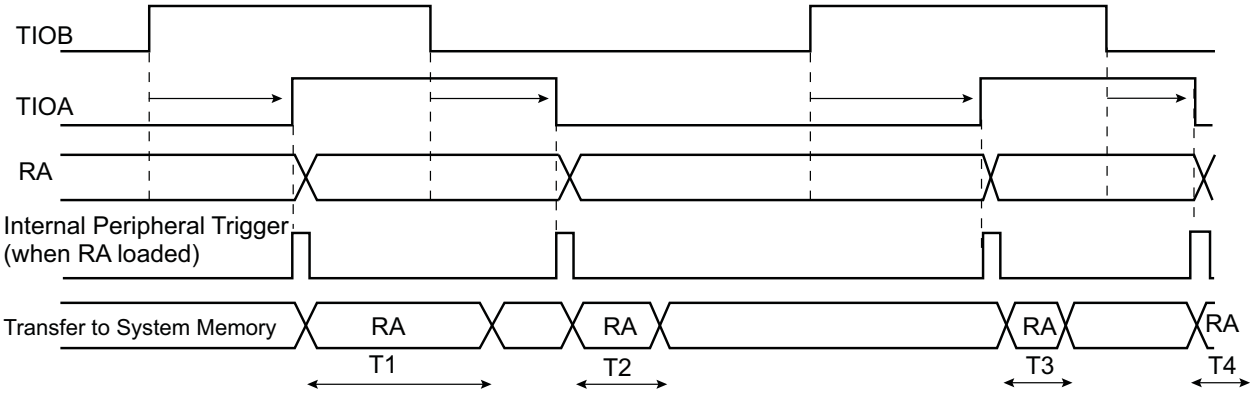
**Figure 53-5. Example of Transfer with DMAC in Capture Mode**

ETRGEDG = 1, LDRA = 1, LDRB = 2, ABETRG = 0




T1,T2,T3,T4 = System Bus load dependent ($t_{min}$ = 8 Peripheral Clocks)

ETRGEDG = 3, LDRA = 3, LDRB = 0, ABETRG = 0



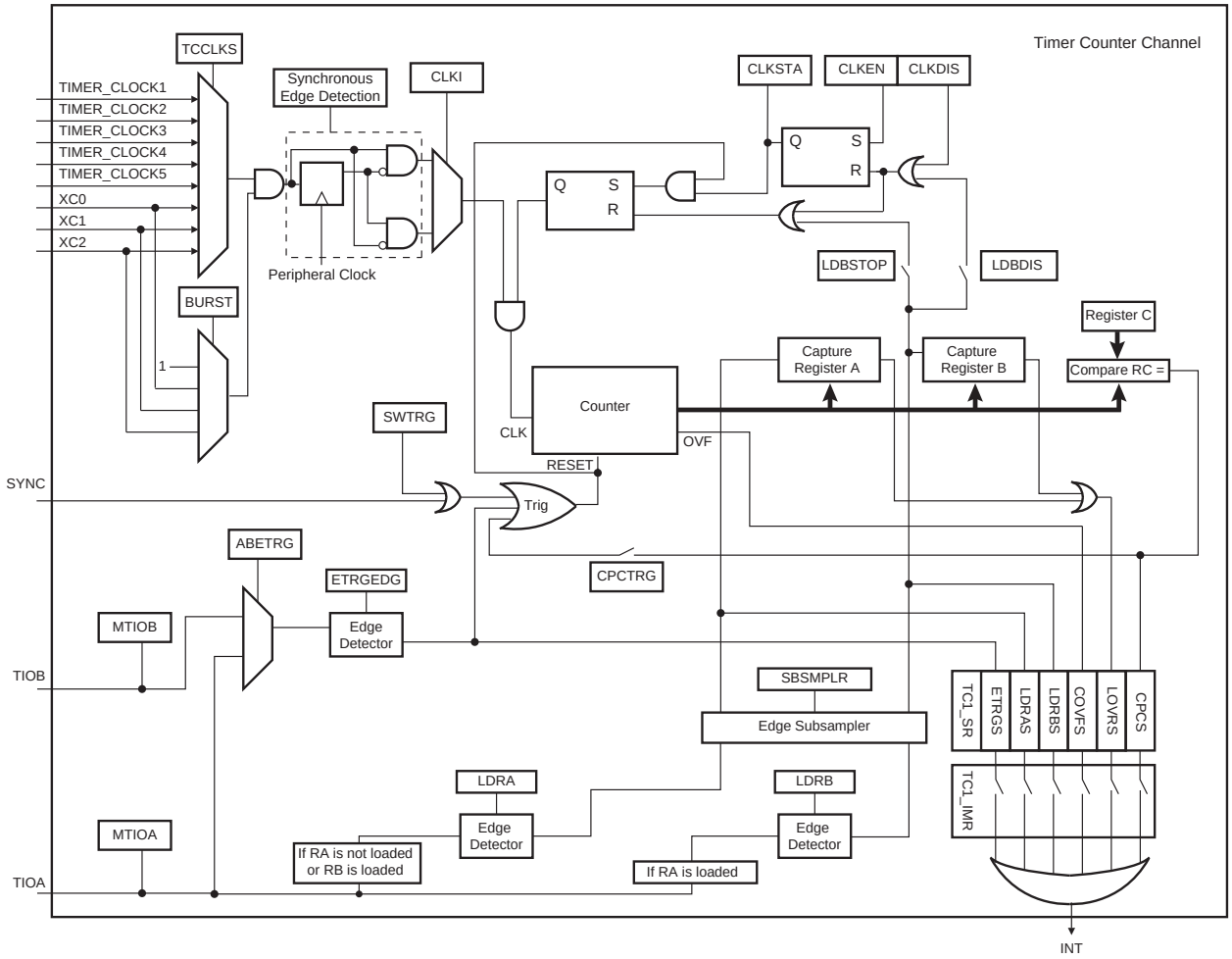

T1,T2,T3,T4 = System Bus load dependent ($t_{min}$ = 8 Peripheral Clocks)

### 53.6.10 Trigger Conditions

In addition to the SYNC signal, the software trigger and the RC compare trigger, an external trigger can be defined.

The ABETRG bit in the TC_CMR selects TIOAx or TIOBx input signal as an external trigger or the trigger signal from the output comparator of the PWM module. The External Trigger Edge Selection parameter (ETRGEDG field in TC_CMR) defines the edge (rising, falling, or both) detected to generate an external trigger. If ETRGEDG = 0 (none), the external trigger is disabled.

**Figure 53-6. Capture Mode**




### 53.6.11 Waveform Mode

Waveform mode is entered by setting the TC_CMRx.WAVE bit.

In Waveform mode, the TC channel generates one or two PWM signals with the same frequency and independently programmable duty cycles, or generates different types of one-shot or repetitive pulses.

In this mode, TIOAx is configured as an output and TIOBx is defined as an output if it is not used as an external event (EEVT parameter in TC_CMR).

Waveform Mode shows the configuration of the TC channel when programmed in Waveform operating mode.

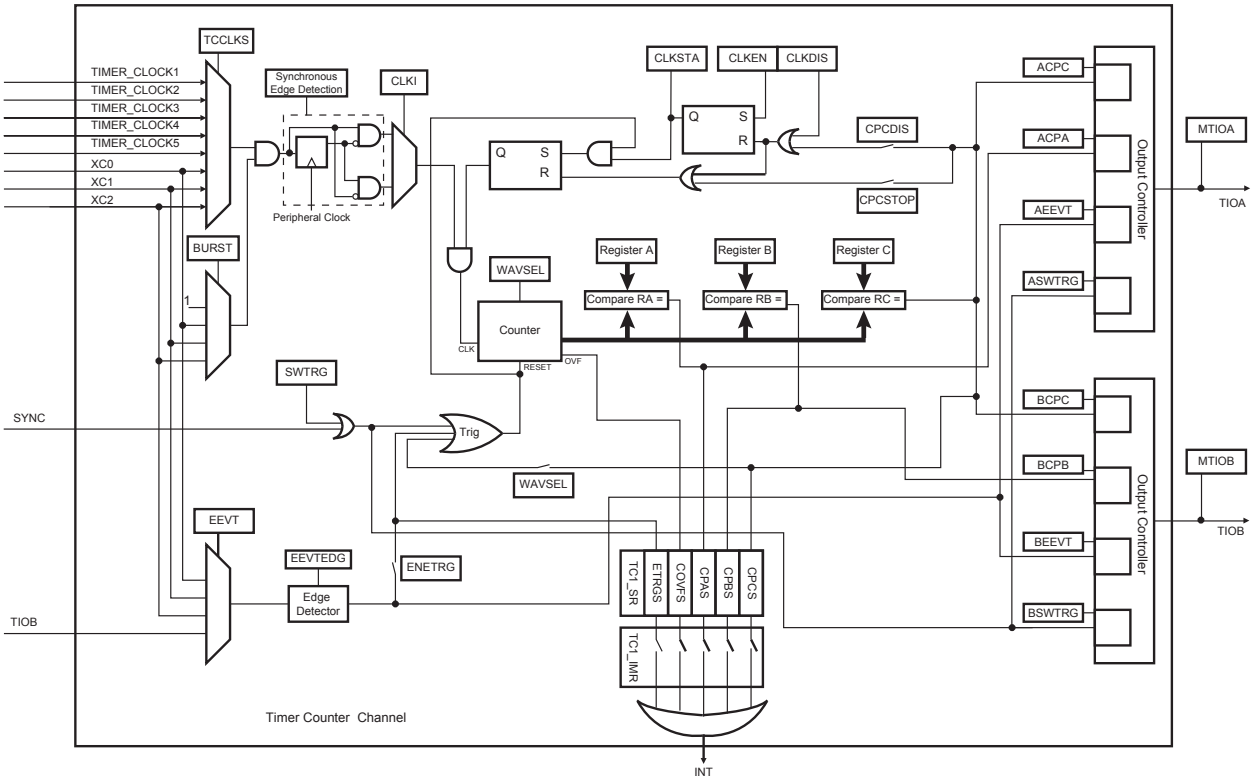
### 53.6.12 Waveform Selection

Depending on the WAVSEL parameter in TC_CMR, the behavior of TC_CV varies.

With any selection, TC_RA, TC_RB and TC_RC can all be used as compare registers.

RA Compare is used to control the TIOAx output, RB Compare is used to control the TIOBx output (if correctly configured) and RC Compare is used to control TIOAx and/or TIOBx outputs.

**Figure 53-7. Waveform Mode**




### 53.6.12.1 WAVSEL = 00

When WAVSEL = 00, the value of TC_CV is incremented from 0 to $2^{32}$-1. Once $2^{32}$-1 has been reached, the value of TC_CV is reset. Incrementation of TC_CV starts again and the cycle continues.

An external event trigger or a software trigger can reset the value of TC_CV. It is important to note that the trigger may occur at any time.

Refer to the figures below.

RC Compare cannot be programmed to generate a trigger in this configuration. At the same time, RC Compare can stop the counter clock (CPCSTOP = 1 in TC_CMR) and/or disable the counter clock (CPCDIS = 1 in TC_CMR).

**Figure 53-8. WAVSEL = 00 without Trigger**

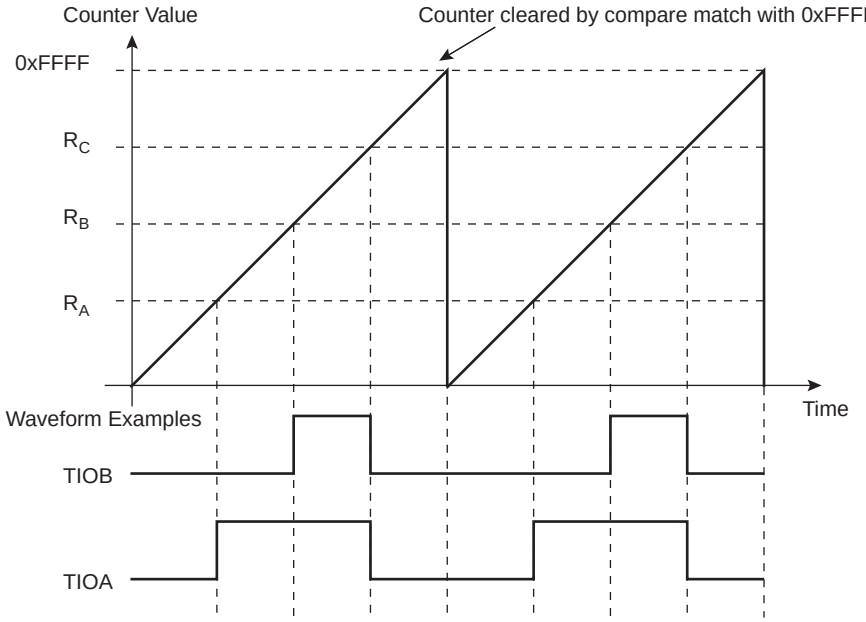
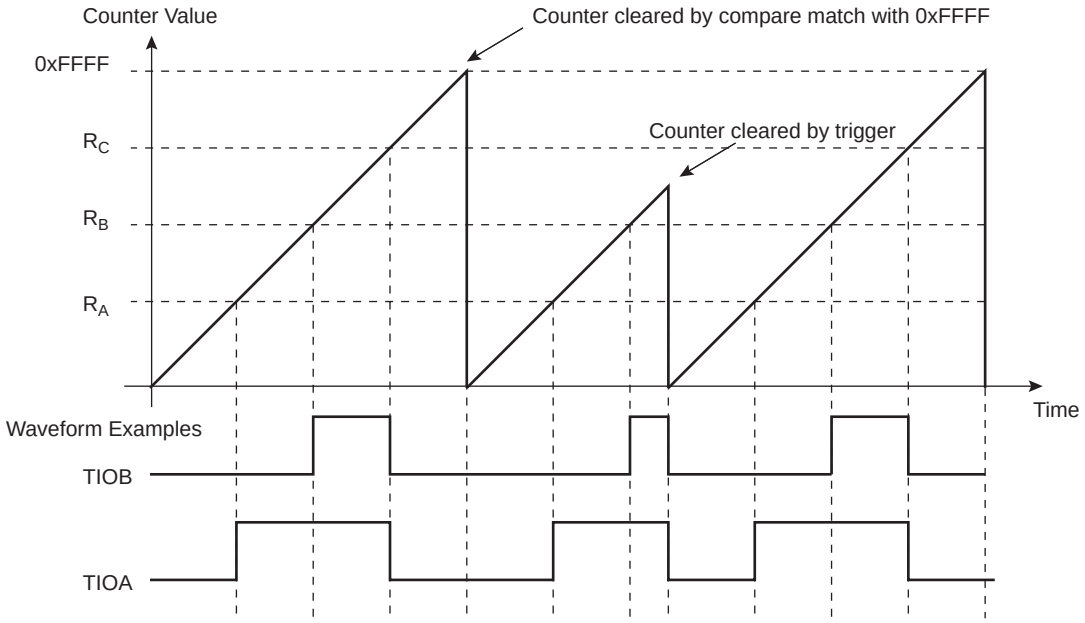



**Figure 53-9. WAVSEL = 00 with Trigger**




#### 53.6.12.2 WAVSEL = 10

When WAVSEL = 10, the value of TC_CV is incremented from 0 to the value of RC, then automatically reset on a RC Compare. Once the value of TC_CV has been reset, it is then incremented and so on.

It is important to note that TC_CV can be reset at any time by an external event or a software trigger if both are programmed correctly.

Refer to the figures below.

In addition, RC Compare can stop the counter clock (CPCSTOP = 1 in TC_CMR) and/or disable the counter clock (CPCDIS = 1 in TC_CMR).

**Figure 53-10. WAVSEL = 10 without Trigger**

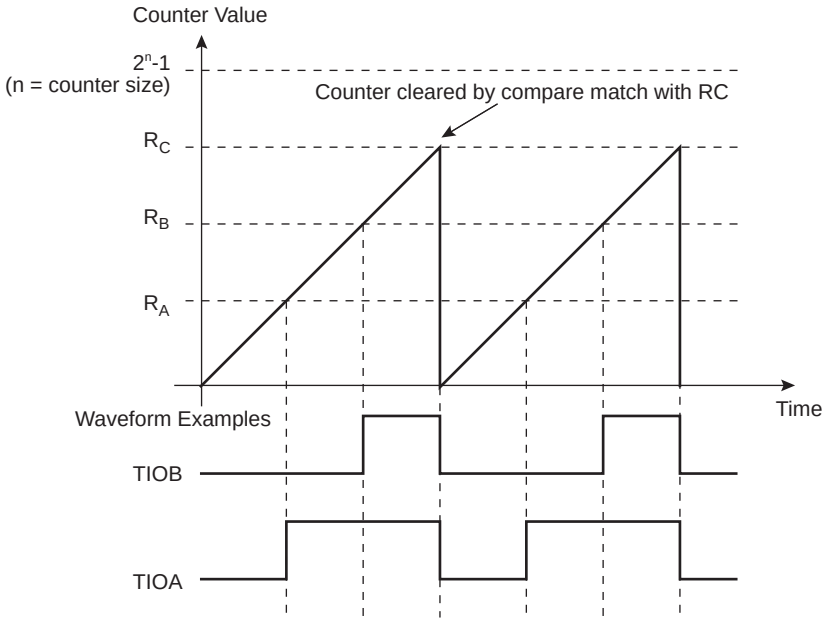
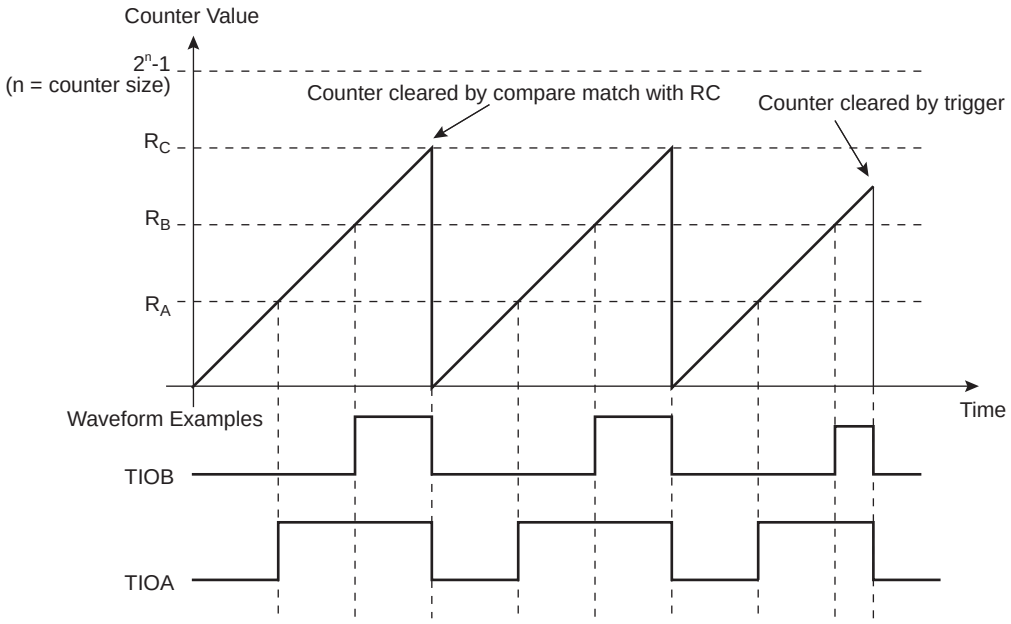



**Figure 53-11. WAVSEL = 10 with Trigger**




#### 53.6.12.3 WAVSEL = 01

When WAVSEL = 01, the value of TC_CV is incremented from 0 to $2^{32}$-1 . Once $2^{32}$-1 is reached, the value of TC_CV is decremented to 0, then reincremented to $2^{32}$-1 and so on.

A trigger such as an external event or a software trigger can modify TC_CV at any time. If a trigger occurs while TC_CV is incrementing, TC_CV then decrements. If a trigger is received while TC_CV is decrementing, TC_CV then increments.

Refer to the figures below.

RC Compare cannot be programmed to generate a trigger in this configuration.

At the same time, RC Compare can stop the counter clock (CPCSTOP = 1) and/or disable the counter clock (CPCDIS = 1).

**Figure 53-12. WAVSEL = 01 without Trigger**

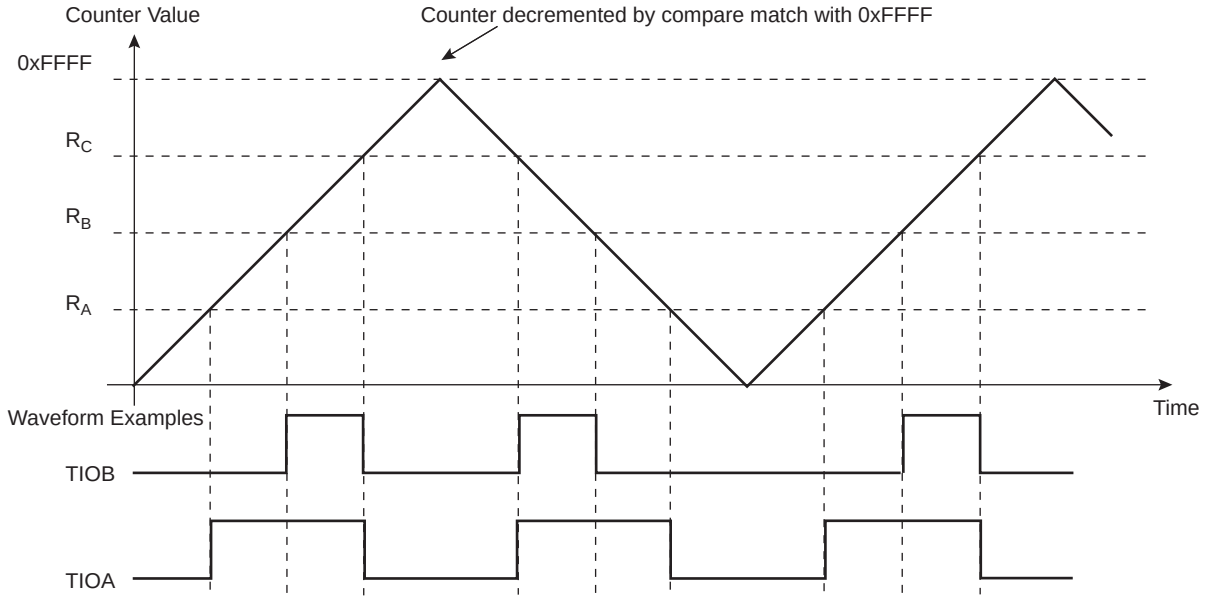
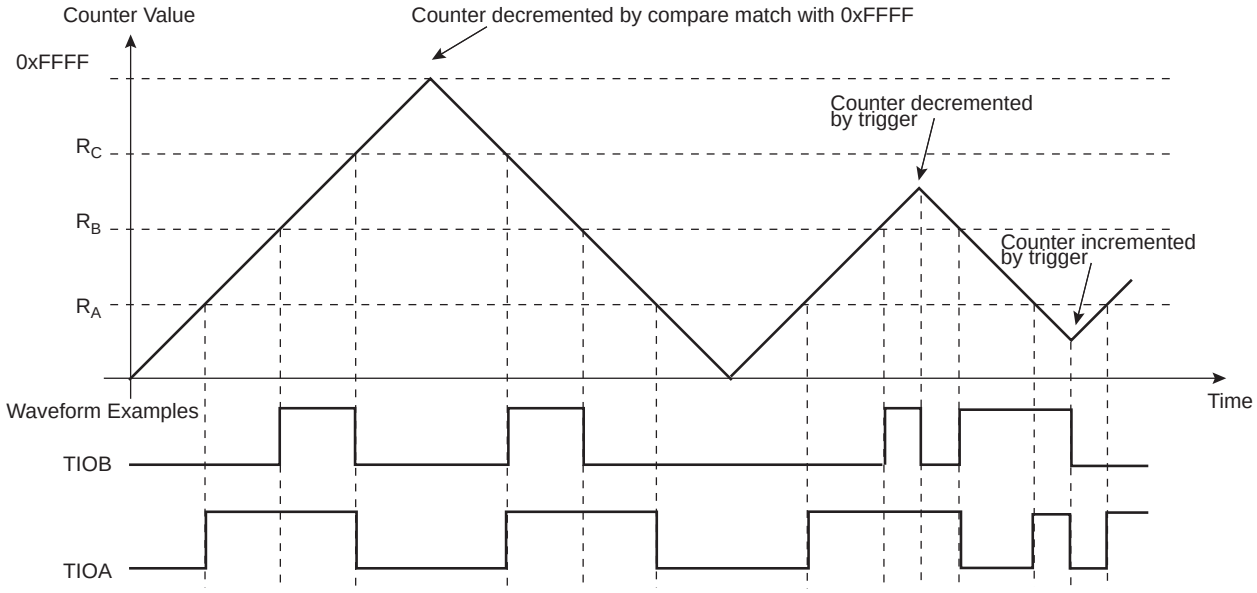



**Figure 53-13. WAVSEL = 01 with Trigger**




#### 53.6.12.4 WAVSEL = 11

When WAVSEL = 11, the value of TC_CV is incremented from 0 to RC. Once RC is reached, the value of TC_CV is decremented to 0, then reincremented to RC and so on.

A trigger such as an external event or a software trigger can modify TC_CV at any time. If a trigger occurs while TC_CV is incrementing, TC_CV then decrements. If a trigger is received while TC_CV is decrementing, TC_CV then increments.

Refer to the figures below.

RC Compare can stop the counter clock (CPCSTOP = 1) and/or disable the counter clock (CPCDIS = 1).

**Figure 53-14. WAVSEL = 11 without Trigger**

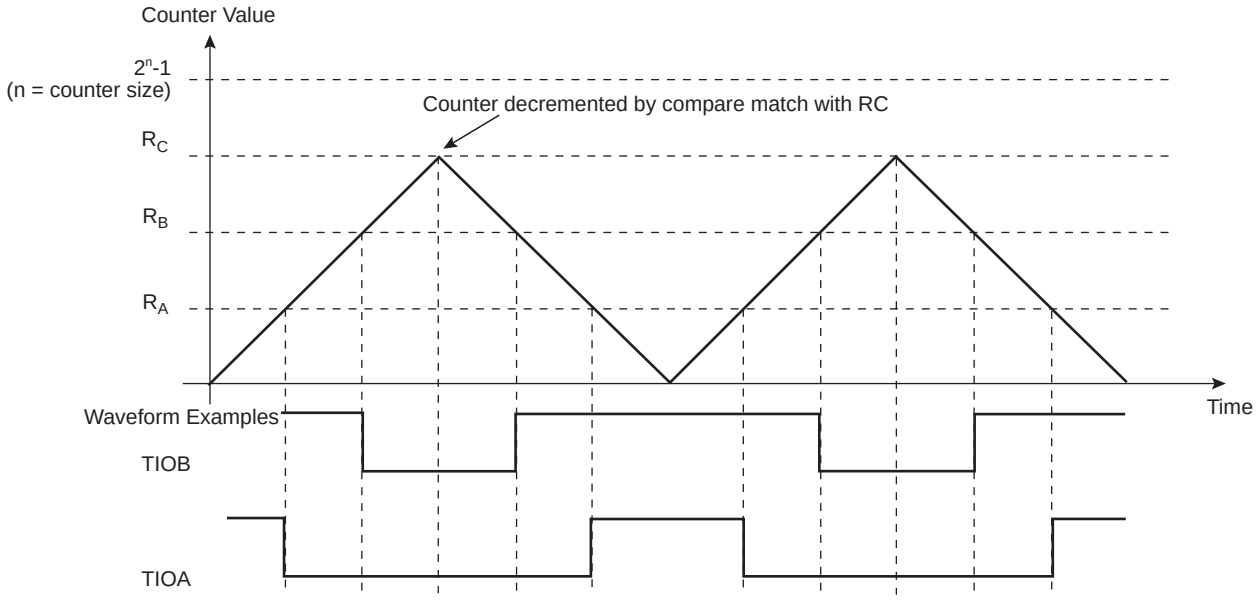
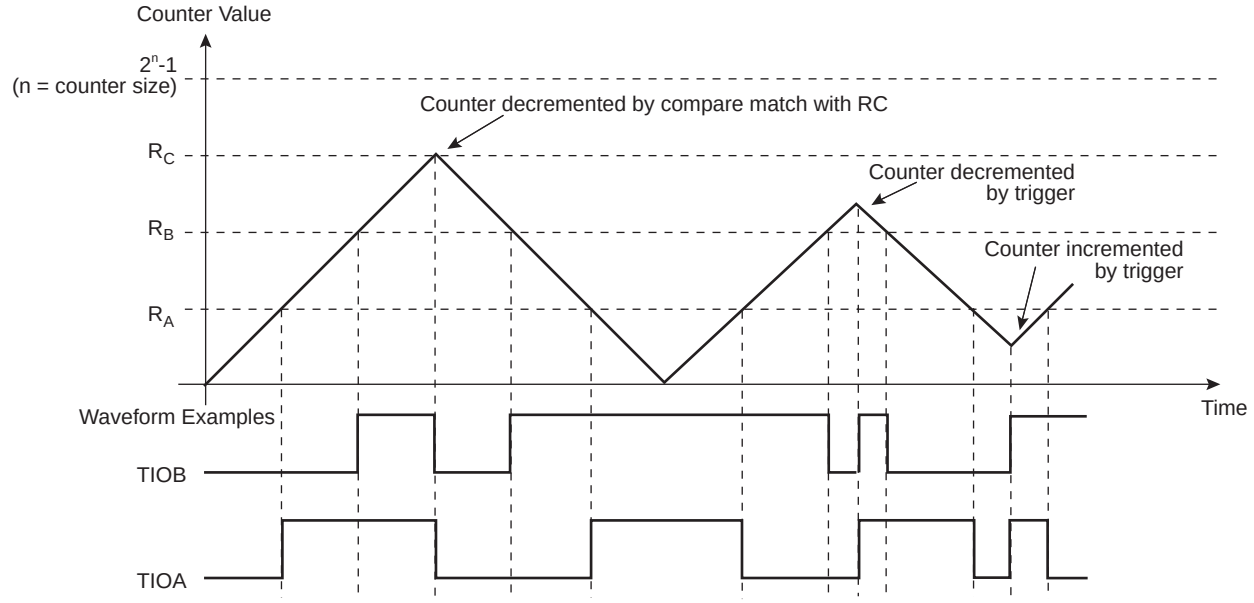



**Figure 53-15. WAVSEL = 11 with Trigger**




### 53.6.13 External Event/Trigger Conditions

An external event can be programmed to be detected on one of the clock sources (XC0, XC1, XC2) or TIOBx. The external event selected can then be used as a trigger.

The event trigger is selected using TC_CMR.EEVT. The trigger edge (rising, falling or both) for each of the possible external triggers is defined in TC_CMR.EEVTEDG. If EEVTEDG is cleared (none), no external event is defined.

If TIOBx is defined as an external event signal (EEVT = 0), TIOBx is no longer used as an output and the compare register B is not used to generate waveforms and subsequently no IRQs. In this case, the TC channel can only generate a waveform on TIOAx.

When an external event is defined, it can be used as a trigger by setting TC_CMR.ENETRG.

As in Capture mode, the SYNC signal and the software trigger are also available as triggers. RC Compare can also be used as a trigger depending on the parameter WAVSEL.

### 53.6.14 Synchronization with PWM

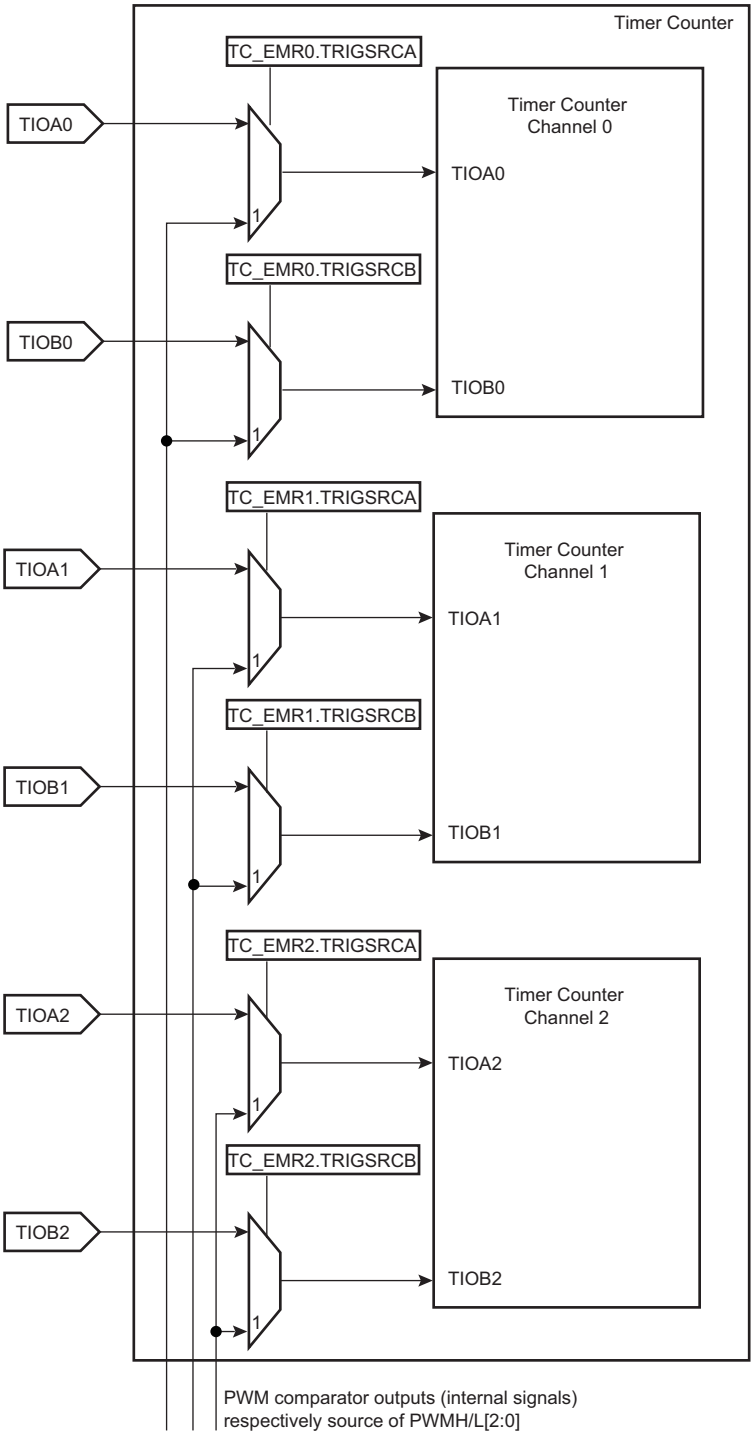
The inputs TIOAx/TIOBx can be bypassed, and thus channel trigger/capture events can be directly driven by the independent PWM module.

PWM comparator outputs (internal signals without dead-time insertion - OCx), respectively source of the PWMH/L[2:0] outputs, are routed to the internal TC inputs. These specific TC inputs are multiplexed with TIOA/B input signal to drive the internal trigger/capture events.

The selection is made in the Extended Mode register (TC_EMR) fields TRIGSRCA and TRIGSRCB (see “TC Extended Mode Register”).

Each channel of the TC module can be synchronized by a different PWM channel as described in the following figure.

**Figure 53-16. Synchronization with PWM**




### 53.6.15 Output Controller

The output controller defines the output level changes on TIOAx and TIOBx following an event. TIOBx control is used only if TIOBx is defined as output (not as an external event).

The following events control TIOAx and TIOBx:

- Software trigger

- External event
- RC compare

RA Compare controls TIOAx, and RB Compare controls TIOBx. Each of these events can be programmed to set, clear or toggle the output as defined in the corresponding parameter in TC_CMR.

### 53.6.16 Quadrature Decoder

#### 53.6.16.1 Description

The quadrature decoder (QDEC) is driven by TIOA0, TIOB0 and TIOB1 input pins and drives the timer counter of channel 0 and 1. Channel 2 can be used as a time base in case of speed measurement requirements (refer to Predefined Connection of the Quadrature Decoder with Timer Counters).

When writing a ‘0’ to TC_BMR.QDEN, the QDEC is bypassed and the IO pins are directly routed to the timer counter function.

TIOA0 and TIOB0 are to be driven by the two dedicated quadrature signals from a rotary sensor mounted on the shaft of the off-chip motor.

A third signal from the rotary sensor can be processed through pin TIOB1 and is typically dedicated to be driven by an index signal if it is provided by the sensor. This signal is not required to decode the quadrature signals PHA, PHB.

TC_CMRx.TCCLKS must be configured to select XC0 input (i.e., 0x101). Field TC0XC0S has no effect as soon as the QDEC is enabled.

Either speed or position/revolution can be measured. Position channel 0 accumulates the edges of PHA, PHB input signals giving a high accuracy on motor position whereas channel 1 accumulates the index pulses of the sensor, therefore the number of rotations. Concatenation of both values provides a high level of precision on motion system position.

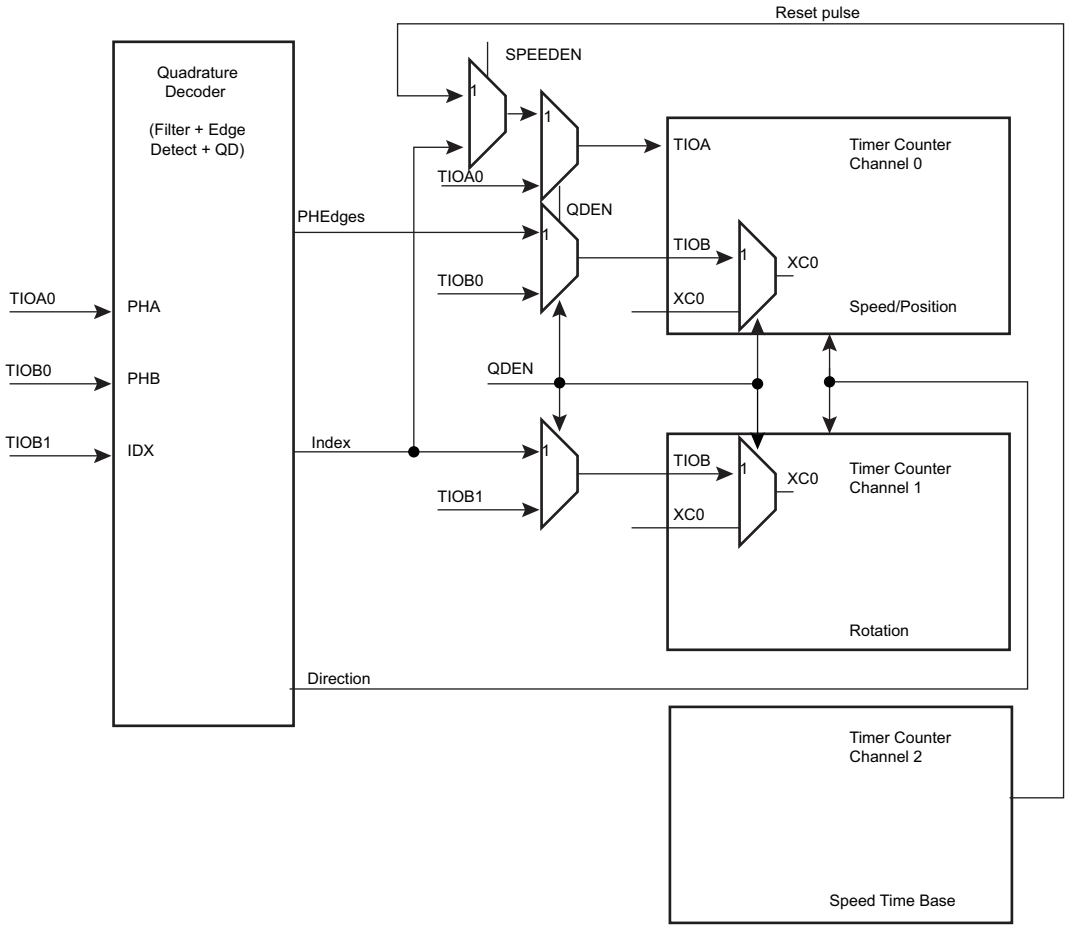
In Speed mode, position cannot be measured but revolution can be measured.

Inputs from the rotary sensor can be filtered prior to downstream processing. Accommodation of input polarity, phase definition and other factors are configurable.

Interruptions can be generated on different events.

A compare function (using TC_RC) is available on channel 0 (speed/position) or channel 1 (rotation) and can generate an interrupt by means of TC_SRx.CPCS.

**Figure 53-17. Predefined Connection of the Quadrature Decoder with Timer Counters**




### 53.6.16.2 Input Preprocessing

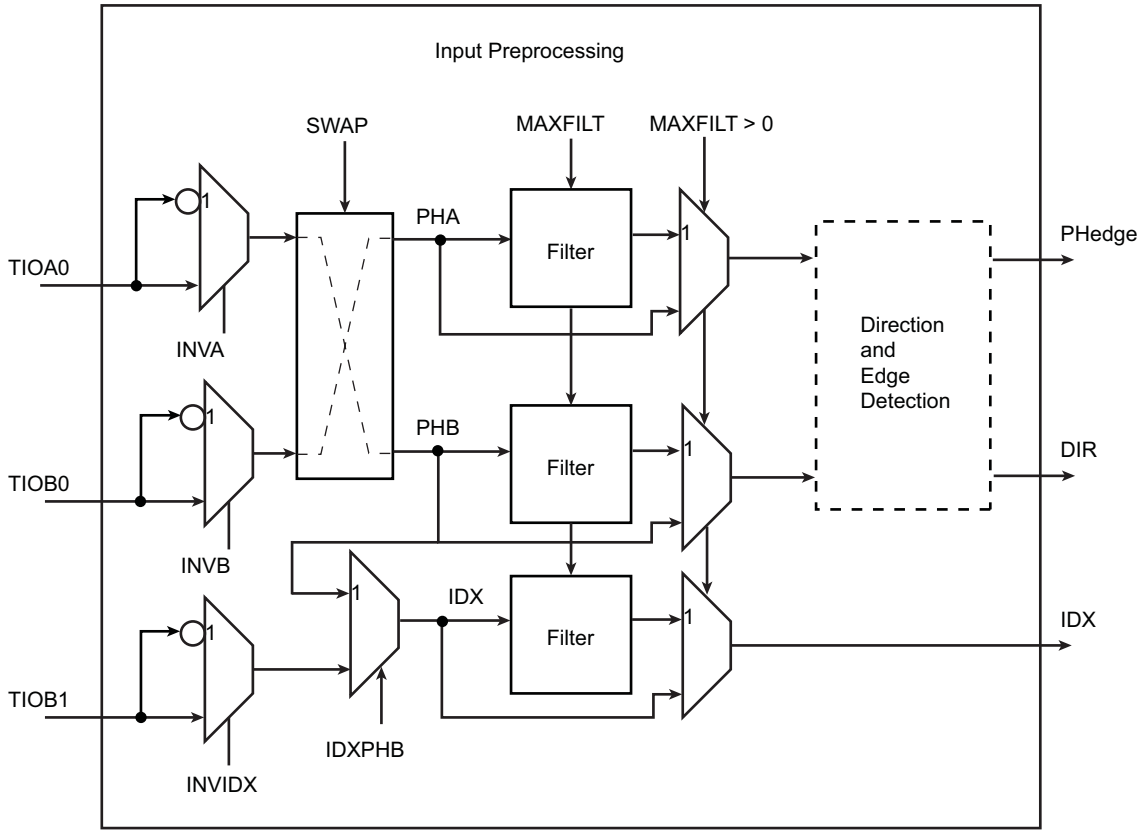
Input preprocessing consists of capabilities to take into account rotary sensor factors such as polarities and phase definition followed by configurable digital filtering.

Each input can be negated and swapping PHA, PHB is also configurable.

TC_BMR. MAXFILT is used to configure a minimum duration for which the pulse is stated as valid. When the filter is active, pulses with a duration lower than (MAXFILT +1) × $t_{peripheral\ clock}$ are not passed to downstream logic.

The value of (MAXFILT +1) × $t_{peripheral\ clock}$ must not be greater than 10% of the minimum pulse on PHA, PHB or index when the rotary encoder speed is at its maximum. This speed depends on the application.

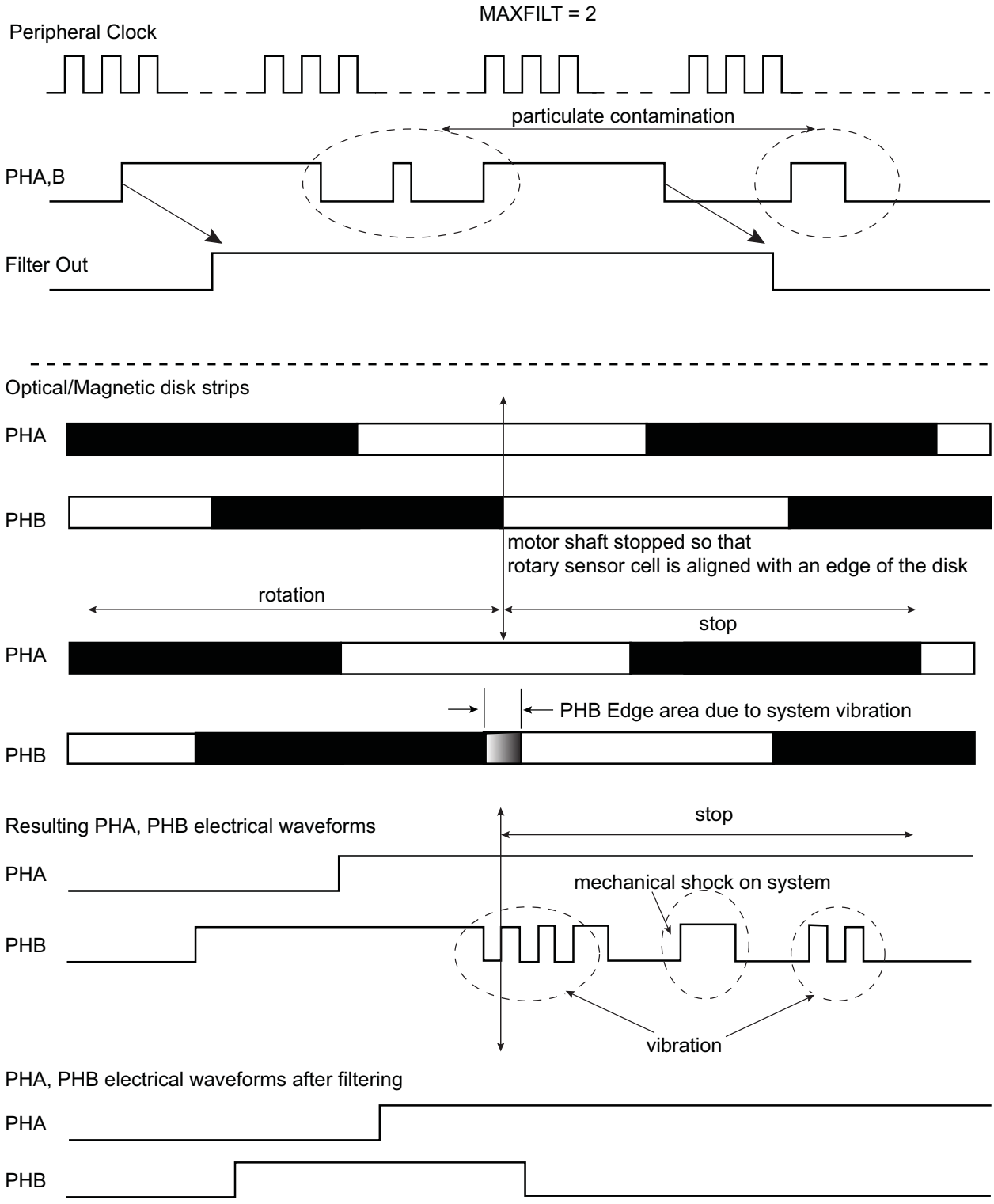
**Figure 53-18. Input Stage**




Input filtering can efficiently remove spurious pulses that might be generated by the presence of particulate contamination on the optical or magnetic disk of the rotary sensor.

Spurious pulses can also occur in environments with high levels of electromagnetic interference. Or, simply if vibration occurs even when rotation is fully stopped and the shaft of the motor is in such a position that the beginning of one of the reflective or magnetic bars on the rotary sensor disk is aligned with the light or magnetic (Hall) receiver cell of the rotary sensor. Any vibration can make the PHA, PHB signals toggle for a short duration.

**Figure 53-19. Filtering Examples**




### 53.6.16.3 Direction Status and Change Detection

After filtering, the quadrature signals are analyzed to extract the rotation direction and edges of the two quadrature signals detected in order to be counted by TC logic downstream.

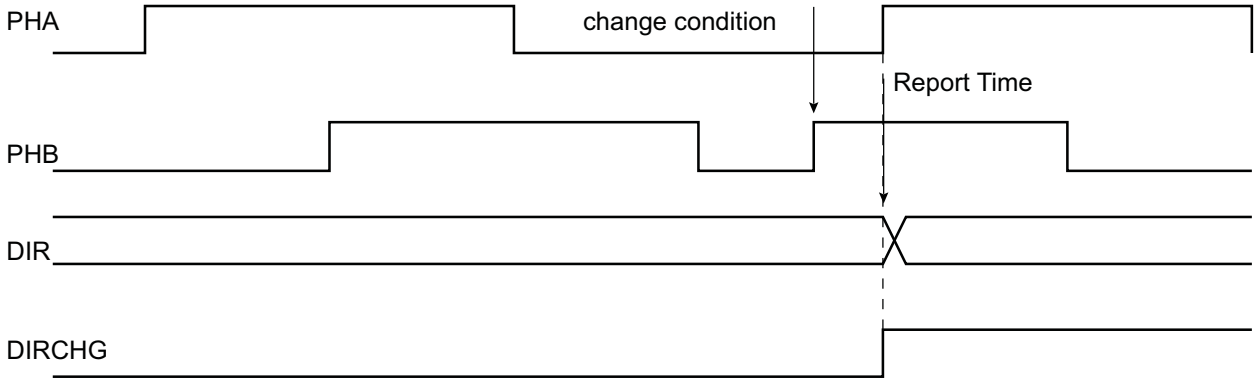
The direction status can be directly read at anytime in the TC_QISR. The polarity of the direction flag status depends on the configuration written in TC_BMR. INVA, INVB, INVIDX, SWAP modify the polarity of DIR flag.

Any change in rotation direction is reported in the TC_QISR and can generate an interrupt.

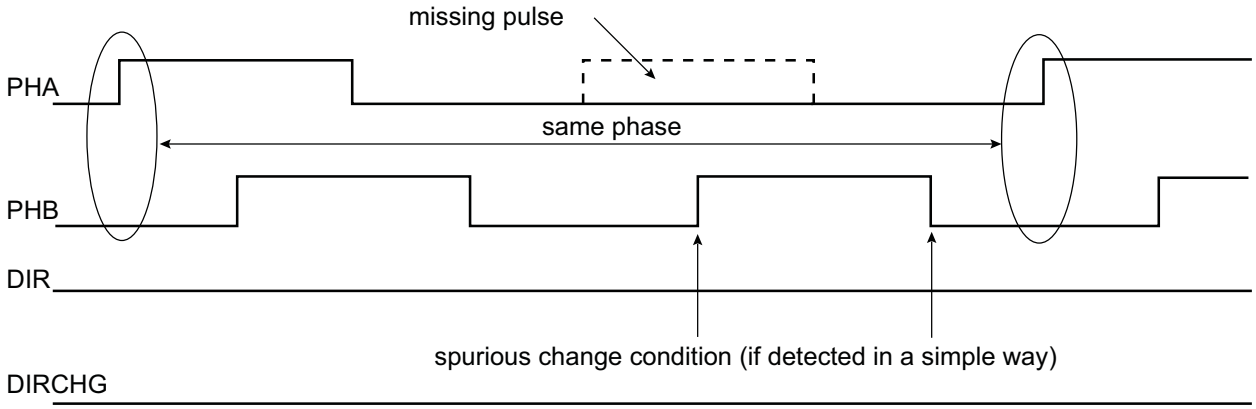
The direction change condition is reported as soon as two consecutive edges on a phase signal have sampled the same value on the other phase signal and there is an edge on the other signal. The two consecutive edges of one phase signal sampling the same value on other phase signal is not sufficient to declare a direction change, as particulate contamination may mask one or more reflective bars on the optical or magnetic disk of the sensor. Refer to the following figure for waveforms.

**Figure 53-20. Rotation Change Detection**

**Direction Change under normal conditions**



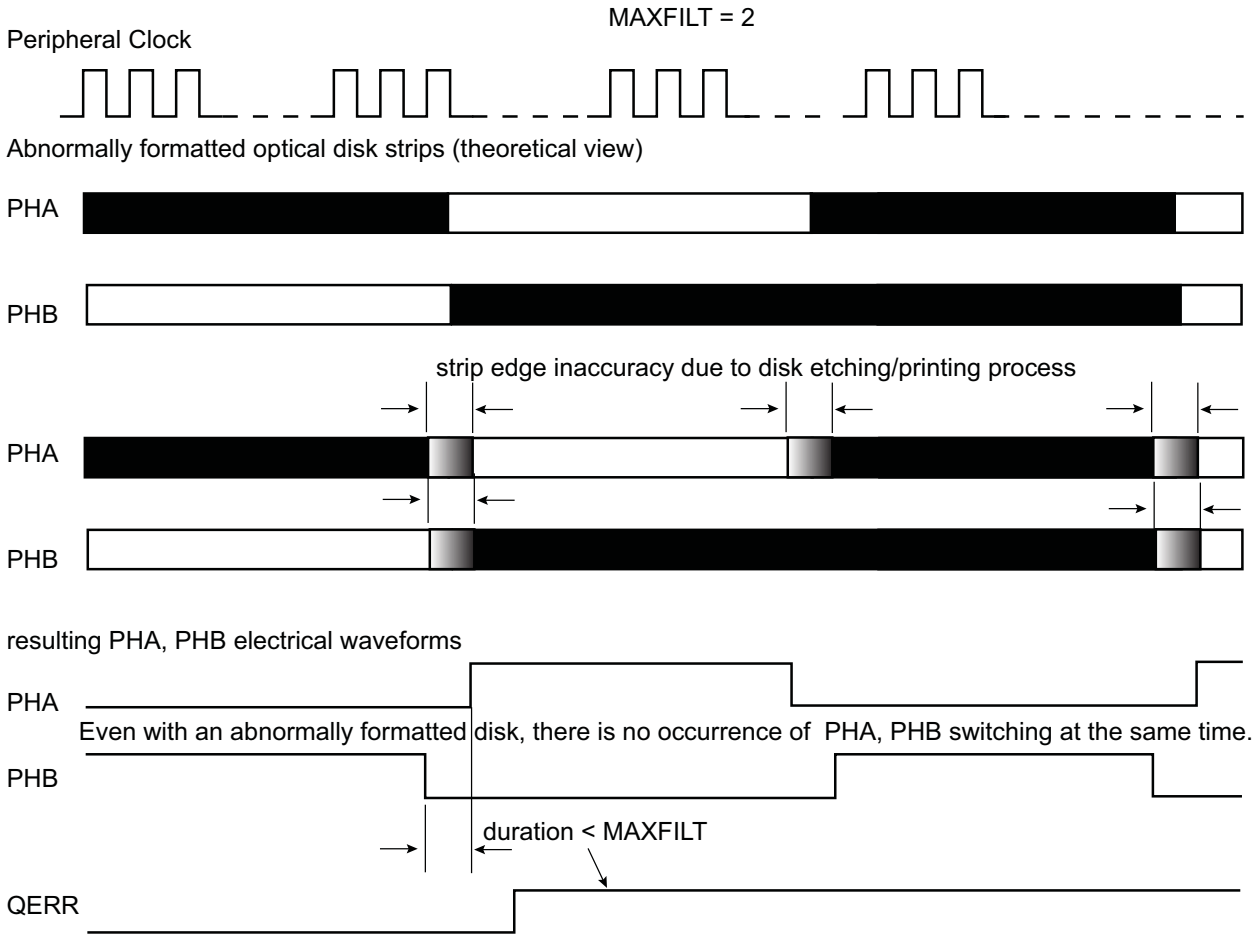

**No direction change due to particulate contamination masking a reflective bar**




The direction change detection is disabled when TC_BMR.QDTRANS is set. In this case, the DIR flag report must not be used.

A quadrature error is also reported by the QDEC via TC_QISR.QERR. This error is reported if the time difference between two edges on PHA, PHB is lower than a predefined value. This predefined value is configurable and corresponds to (TC_BMR.MAXFILT + 1) × $t_{peripheral\ clock}$ ns. After being filtered, there is no reason to have two edges closer than (TC_BMR.MAXFILT + 1) × $t_{peripheral\ clock}$ ns under normal mode of operation.

**Figure 53-21. Quadrature Error Detection**




MAXFILT must be tuned according to several factors such as the peripheral clock frequency, type of rotary sensor and rotation speed to be achieved.

#### 53.6.16.4 Position and Rotation Measurement

When TC_BMR.POSEN is set, the motor axis position is processed on channel 0 (by means of the PHA, PHB edge detections) and the number of motor revolutions are recorded on channel 1 if the IDX signal is provided on the TIOB1 input. If no IDX signal is available, the internal counter can be cleared for each revolution if the number of counts per revolution is configured in TC_RC0.RC and the TC_CMR.CPCTRG bit is written to ‘1’. The position measurement can be read in the TC_CV0 register and the rotation measurement can be read in the TC_CV1 register.

Channel 0 and 1 must be configured in Capture mode (TC_CMR0.WAVE = 0). ‘Rising edge’ must be selected as the External Trigger Edge (TC_CMR.ETRGEDG = 0x01) and ‘TIOAx’ must be selected as the External Trigger (TC_CMR.ABETRG = 0x1). The process must be started by configuring TC_CCR.CLKEN and TC_CCR.SWTRG.

In parallel, the number of edges are accumulated on TC channel 0 and can be read on the TC_CV0 register.

Therefore, the accurate position can be read on both TC_CV registers and concatenated to form a 32-bit word.

The TC channel 0 is cleared for each increment of IDX count value.

Depending on the quadrature signals, the direction is decoded and allows to count up or down in TC channels 0 and 1. The direction status is reported on TC_QISR.

#### 53.6.16.5 Speed Measurement

When TC_BMR.SPEEDEN is set, the speed measure is enabled on channel 0.

A time base must be defined on channel 2 by writing the TC_RC2 period register. Channel 2 must be configured in Waveform mode (WAVE bit set) in TC_CMR2. The WAVSEL field must be defined with 0x10 to clear the counter by comparison and matching with TC_RC value. Field ACPC must be defined at 0x11 to toggle TIOAx output.

This time base is automatically fed back to TIOAx of channel 0 when QDEN and SPEEDEN are set.

Channel 0 must be configured in Capture mode (WAVE = 0 in TC_CMR0). TC_CMR0.ABETRG must be configured at 1 to select TIOAx as a trigger for this channel.

EDGTRG must be set to 0x01, to clear the counter on a rising edge of the TIOAx signal and field LDRA must be set accordingly to 0x01, to load TC_RA0 at the same time as the counter is cleared (LDRB must be set to 0x01). As a consequence, at the end of each time base period the differentiation required for the speed calculation is performed.

The process must be started by configuring bits CLKEN and SWTRG in the TC_CCR.

The speed can be read on field RA in TC_RA0.

Channel 1 can still be used to count the number of revolutions of the motor.

#### 53.6.16.6 Detecting a Missing Index Pulse

To detect a missing index pulse due contamination, dust, etc., the TC_SR0.CPCS flag can be used. It is also possible to assert the interrupt line if the TC_SR0.CPCS flag is enabled as a source of the interrupt by writing a ‘1’ to TC_IER0.CPCS.

The TC_RC0.RC field must be written with the nominal number of counts per revolution provided by the rotary encoder, plus a margin to eliminate potential noise (e.g., if nominal count per revolution is 1024, then TC_RC0.RC=1026).

If the index pulse is missing, the timer value is not cleared and the nominal value is exceeded, then the comparator on the RC triggers an event, TC_SR0.CPCS=1, and the interrupt line is asserted if TC_IER0.CPCS=1.

The missing index pulse detection is only valid if the bit TC_QISR.DIRCHG=0.

#### 53.6.16.7 Detecting Contamination/Dust at Rotary Encoder Low Speed

The contamination/dust that can be filtered when the rotary encoder speed is high may not be filtered at low speed, thus creating unsollicited direction change, etc.

At low speed, even a minor contamination may appear as a long pulse, and thus not filtered and processed as a standard quadrature encoder pulse.

This contamination can be detected by using the similar method as the missing index detection.

A contamination exists on a phase line if TC_SR.CPCS = 1 and TC_QISR.DIRCHG = 1 when there is no sollicited change of direction.

#### 53.6.16.8 Missing Pulse Detection and Autocorrection

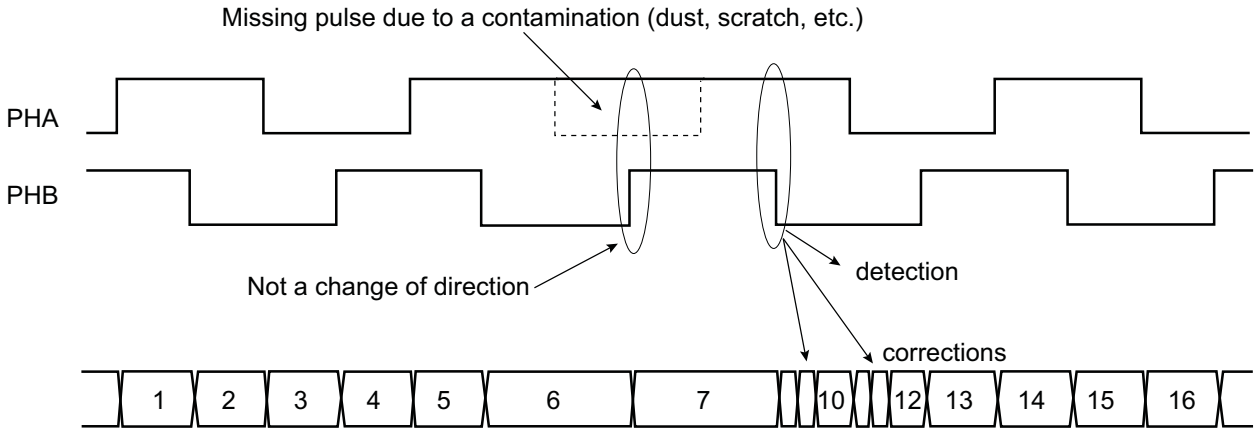
The QDEC is equipped with a circuitry which detects and corrects some errors that may result from contamination on optical disks or other materials producing the quadrature phase signals.

The detection and autocorrection only works if the Count mode is configured for both phases (EDGPHA = 1 in TC_BMR) and is enabled (AUTOC = 1 in TC_BMR).

If a pulse is missing on a phase signal, it is automatically detected and the pulse count reported in the CV field of the TC_CV0/1 is automatically corrected.

There is no detection if both phase signals are affected at the same location on the device providing the quadrature signals because the detection requires a valid phase signal to detect the contamination on the other phase signal.

**Figure 53-22. Detection and Autocorrection of Missing Pulses**




If a quadrature device is undamaged, the number of pulses counted for a predefined period of time must be the same with or without detection and autocorrection feature.

Therefore, if the measurement results differ, a contamination exists on the device producing the quadrature signals.

This does not substitute the measurements of the number of pulses between two index pulses (if available) but provides a complementary method to detect damaged quadrature devices.

When the device providing quadrature signals is severely damaged, potentially leading to a number of consecutive missing pulses greater than 1, the downstream processing may be affected. It is possible to define the maximum admissible number of consecutive missing pulses before issuing a Missing Pulse Error flag (MPE in TC_QISR). The threshold triggering an MPE flag report can be configured in TC_BMR.MAXCMP. If the field MAXCMP is cleared, MPE never rises. The flag MAXCMP can trigger an interrupt while the QDEC is operating, thus providing a real time report of a potential problem on the quadrature device.

### 53.6.17 2-bit Gray Up/Down Counter for Stepper Motor

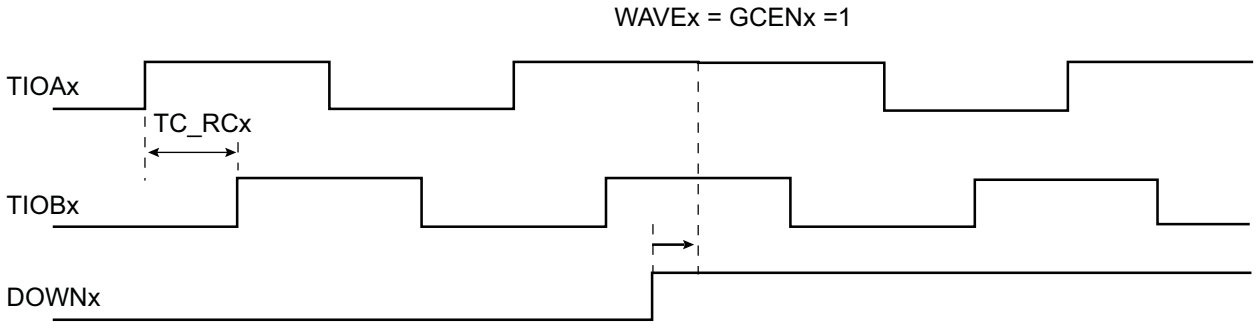
Each channel can be independently configured to generate a 2-bit Gray count waveform on corresponding TIOAx, TIOBx outputs by means of TC_SMMRx.GCEN.

Up or Down count can be defined by writing TC_SMMRx.DOWN.

It is mandatory to configure the channel in Waveform mode in the TC_CMR.

The period of the counters can be programmed in TC_RCx.

**Figure 53-23. 2-bit Gray Up/Down Counter**




### 53.6.18 Fault Mode

At any time, the TC_RCx registers can be used to perform a comparison on the respective current channel counter value (TC_CVx) with the value of TC_RCx register.

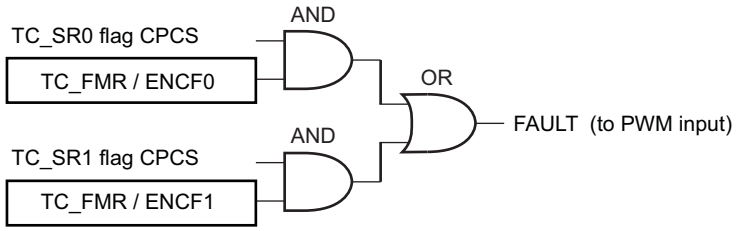
The CPCSx flags can be set accordingly and an interrupt can be generated.

This interrupt is processed but requires an unpredictable amount of time to be achieve the required action.

It is possible to trigger the FAULT output of the TIMER1 with CPCS from TC_SR0 and/or CPCS from TC_SR1. Each source can be independently enabled/disabled in the TC_FMR.

This can be useful to detect an overflow on speed and/or position when QDEC is processed and to act immediately by using the FAULT output.

**Figure 53-24. Fault Output Generation**




### 53.6.19 Register Write Protection

To prevent any single software error from corrupting TC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the TC Write Protection Mode Register (TC_WPMR).

The Timer Counter clock of the first channel must be enabled to access TC_WPMR.

The following registers can be write-protected when WPEN is set:

- TC Block Mode Register
- TC Channel Mode Register Capture Mode
- TC Channel Mode Register Waveform Mode
- TC Fault Mode Register
- TC Stepper Motor Mode Register
- TC Register A
- TC Register B
- TC Register C
- TC Extended Mode Register

## 53.7 Register Summary

**Note:** The register TC_CMR has two modes, Capture Mode and Waveform Mode. In this register summary, both modes are displayed

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | TC_CCR0 | 7:0 | | | | | | SWTRG | CLKDIS | CLKEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | TC_CMR0 | 7:0 | LDBDIS | LDBSTOP | BURST[1:0] | | CLKI | TCCLKS[2:0] | | |
| | | 15:8 | WAVE | CPCTRG | | | | ABETRG | ETRGEDG[1:0] | |
| | | 23:16 | | SBSMPLR[2:0] | | | LDRB[1:0] | | LDRA[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x04 | TC_CMR0 | 7:0 | CPCDIS | CPCSTOP | BURST[1:0] | | CLKI | TCCLKS[2:0] | | |
| | | 15:8 | WAVE | WAVSEL[1:0] | | ENETRG | EEVT[1:0] | | EEVTEDG[1:0] | |
| | | 23:16 | ASWTRG[1:0] | | AEEVT[1:0] | | ACPC[1:0] | | ACPA[1:0] | |
| | | 31:24 | BSWTRG[1:0] | | BEEVT[1:0] | | BCPC[1:0] | | BCPB[1:0] | |
| 0x08 | TC_SMMR0 | 7:0 | | | | | | | DOWN | GCEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | TC_RAB0 | 7:0 | RAB[7:0] | | | | | | | |
| | | 15:8 | RAB[15:8] | | | | | | | |
| | | 23:16 | RAB[23:16] | | | | | | | |
| | | 31:24 | RAB[31:24] | | | | | | | |
| 0x10 | TC_CV0 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | CV[31:24] | | | | | | | |
| 0x14 | TC_RA0 | 7:0 | RA[7:0] | | | | | | | |
| | | 15:8 | RA[15:8] | | | | | | | |
| | | 23:16 | RA[23:16] | | | | | | | |
| | | 31:24 | RA[31:24] | | | | | | | |
| 0x18 | TC_RB0 | 7:0 | RB[7:0] | | | | | | | |
| | | 15:8 | RB[15:8] | | | | | | | |
| | | 23:16 | RB[23:16] | | | | | | | |
| | | 31:24 | RB[31:24] | | | | | | | |
| 0x1C | TC_RC0 | 7:0 | RC[7:0] | | | | | | | |
| | | 15:8 | RC[15:8] | | | | | | | |
| | | 23:16 | RC[23:16] | | | | | | | |
| | | 31:24 | RC[31:24] | | | | | | | |
| 0x20 | TC_SR0 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | MTIOB | MTIOA | CLKSTA |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x24 | TC_IER0 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x28 | TC_IDR0 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x2C | TC_IMR0 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x30 | TC_EMR0 | 7:0 | | | TRIGSRCB[1:0] | | | | TRIGSRCA[1:0] | |
| | | 15:8 | | | | | | | | NODIVCLK |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x34 ... 0x3F | Reserved | | | | | | | | | |
| 0x40 | TC_CCR1 | 7:0 | | | | | | SWTRG | CLKDIS | CLKEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x44 | TC_CMR1 | 7:0 | LDBDIS | LDBSTOP | BURST[1:0] | | CLKI | TCCLKS[2:0] | | |
| | | 15:8 | WAVE | CPCTRG | | | | ABETRG | ETRGEDG[1:0] | |
| | | 23:16 | | SBSMPLR[2:0] | | | LDRB[1:0] | | LDRA[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x44 | TC_CMR1 | 7:0 | CPCDIS | CPCSTOP | BURST[1:0] | | CLKI | TCCLKS[2:0] | | |
| | | 15:8 | WAVE | WAVSEL[1:0] | | ENETRG | EEVT[1:0] | | EEVTEDG[1:0] | |
| | | 23:16 | ASWTRG[1:0] | | AEEVT[1:0] | | ACPC[1:0] | | ACPA[1:0] | |
| | | 31:24 | BSWTRG[1:0] | | BEEVT[1:0] | | BCPC[1:0] | | BCPB[1:0] | |
| 0x48 | TC_SMMR1 | 7:0 | | | | | | | DOWN | GCEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x4C | TC_RAB1 | 7:0 | RAB[7:0] | | | | | | | |
| | | 15:8 | RAB[15:8] | | | | | | | |
| | | 23:16 | RAB[23:16] | | | | | | | |
| | | 31:24 | RAB[31:24] | | | | | | | |
| 0x50 | TC_CV1 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | CV[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x54 | TC_RA1 | 7:0 | RA[7:0] | | | | | | | |
| | | 15:8 | RA[15:8] | | | | | | | |
| | | 23:16 | RA[23:16] | | | | | | | |
| | | 31:24 | RA[31:24] | | | | | | | |
| 0x58 | TC_RB1 | 7:0 | RB[7:0] | | | | | | | |
| | | 15:8 | RB[15:8] | | | | | | | |
| | | 23:16 | RB[23:16] | | | | | | | |
| | | 31:24 | RB[31:24] | | | | | | | |
| 0x5C | TC_RC1 | 7:0 | RC[7:0] | | | | | | | |
| | | 15:8 | RC[15:8] | | | | | | | |
| | | 23:16 | RC[23:16] | | | | | | | |
| | | 31:24 | RC[31:24] | | | | | | | |
| 0x60 | TC_SR1 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | MTIOB | MTIOA | CLKSTA |
| | | 31:24 | | | | | | | | |
| 0x64 | TC_IER1 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x68 | TC_IDR1 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x6C | TC_IMR1 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x70 | TC_EMR1 | 7:0 | | | TRIGSRCB[1:0] | | | | TRIGSRCA[1:0] | |
| | | 15:8 | | | | | | | | NODIVCLK |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x74 ... 0x7F | Reserved | | | | | | | | | |
| 0x80 | TC_CCR2 | 7:0 | | | | | | SWTRG | CLKDIS | CLKEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x84 | TC_CMR2 | 7:0 | LDBDIS | LDBSTOP | BURST[1:0] | | CLKI | TCCLKS[2:0] | | |
| | | 15:8 | WAVE | CPCTRG | | | | ABETRG | ETRGEDG[1:0] | |
| | | 23:16 | | SBSMPLR[2:0] | | | LDRB[1:0] | | LDRA[1:0] | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x84 | TC_CMR2 | 7:0 | CPCDIS | CPCSTOP | BURST[1:0] | | CLKI | TCCLKS[2:0] | | |
| | | 15:8 | WAVE | WAVSEL[1:0] | | ENETRG | EEVT[1:0] | | EEVTEDG[1:0] | |
| | | 23:16 | ASWTRG[1:0] | | AEEVT[1:0] | | ACPC[1:0] | | ACPA[1:0] | |
| | | 31:24 | BSWTRG[1:0] | | BEEVT[1:0] | | BCPC[1:0] | | BCPB[1:0] | |
| 0x88 | TC_SMMR2 | 7:0 | | | | | | | DOWN | GCEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x8C | TC_RAB2 | 7:0 | RAB[7:0] | | | | | | | |
| | | 15:8 | RAB[15:8] | | | | | | | |
| | | 23:16 | RAB[23:16] | | | | | | | |
| | | 31:24 | RAB[31:24] | | | | | | | |
| 0x90 | TC_CV2 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | CV[31:24] | | | | | | | |
| 0x94 | TC_RA2 | 7:0 | RA[7:0] | | | | | | | |
| | | 15:8 | RA[15:8] | | | | | | | |
| | | 23:16 | RA[23:16] | | | | | | | |
| | | 31:24 | RA[31:24] | | | | | | | |
| 0x98 | TC_RB2 | 7:0 | RB[7:0] | | | | | | | |
| | | 15:8 | RB[15:8] | | | | | | | |
| | | 23:16 | RB[23:16] | | | | | | | |
| | | 31:24 | RB[31:24] | | | | | | | |
| 0x9C | TC_RC2 | 7:0 | RC[7:0] | | | | | | | |
| | | 15:8 | RC[15:8] | | | | | | | |
| | | 23:16 | RC[23:16] | | | | | | | |
| | | 31:24 | RC[31:24] | | | | | | | |
| 0xA0 | TC_SR2 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | MTIOB | MTIOA | CLKSTA |
| | | 31:24 | | | | | | | | |
| 0xA4 | TC_IER2 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xA8 | TC_IDR2 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xAC | TC_IMR2 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xB0 | TC_EMR2 | 7:0 | | | TRIGSRCB[1:0] | | | | TRIGSRCA[1:0] | |
| | | 15:8 | | | | | | | | NODIVCLK |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xB4 ... 0xBF | Reserved | | | | | | | | | |
| 0xC0 | TC_BCR | 7:0 | | | | | | | | SYNC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xC4 | TC_BMR | 7:0 | | | TC2XC2S[1:0] | | TC1XC1S[1:0] | | TC0XC0S[1:0] | |
| | | 15:8 | INVIDX | INVB | INVA | EDGPHA | QDTRANS | SPEEDEN | POSEN | QDEN |
| | | 23:16 | MAXFILT[3:0] | | | | | AUTOC | IDXPHB | SWAP |
| | | 31:24 | | | MAXCMP[3:0] | | | | MAXFILT[5:4] | |
| 0xC8 | TC_QIER | 7:0 | | | | | MPE | QERR | DIRCHG | IDX |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xCC | TC_QIDR | 7:0 | | | | | MPE | QERR | DIRCHG | IDX |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xD0 | TC_QIMR | 7:0 | | | | | MPE | QERR | DIRCHG | IDX |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xD4 | TC_QISR | 7:0 | | | | | MPE | QERR | DIRCHG | IDX |
| | | 15:8 | | | | | | | | DIR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xD8 | TC_FMR | 7:0 | | | | | | | ENCF1 | ENCF0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xDC ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | TC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |

### 53.7.1 TC Channel Control Register

**Name:** TC_CCRx
**Offset:** 0x00 + x*0x40 [x=0..2]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SWTRG | CLKDIS | CLKEN |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

**Bit 2 – SWTRG** Software Trigger Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | A software trigger is performed: the counter is reset and the clock is started. |

**Bit 1 – CLKDIS** Counter Clock Disable Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the clock. |

**Bit 0 – CLKEN** Counter Clock Enable Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the clock if CLKDIS is not 1. |

### 53.7.2 TC Channel Mode Register: Capture Mode

**Name:** TC_CMRx
**Offset:** 0x04 + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read/Write

This register can be written only if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | SBSMPLR[2:0] | | | LDRB[1:0] | | LDRA[1:0] | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WAVE | CPCTRG | | | | ABETRG | ETRGEDG[1:0] | |
| Access | R/W | R/W | | | | R/W | R/W | R/W |
| Reset | 0 | 0 | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LDBDIS | LDBSTOP | BURST[1:0] | | CLKI | TCCLKS[2:0] | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 22:20 – SBSMPLR[2:0]** Loading Edge Subsampling Ratio

| Value | Name | Description |
|---|---|---|
| 0 | ONE | Load a Capture register each selected edge. |
| 1 | HALF | Load a Capture register every 2 selected edges. |
| 2 | FOURTH | Load a Capture register every 4 selected edges. |
| 3 | EIGHTH | Load a Capture register every 8 selected edges. |
| 4 | SIXTEENTH | Load a Capture register every 16 selected edges. |

**Bits 19:18 – LDRB[1:0]** RB Loading Edge Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | RISING | Rising edge of TIOAx |
| 2 | FALLING | Falling edge of TIOAx |
| 3 | EDGE | Each edge of TIOAx |

**Bits 17:16 – LDRA[1:0]** RA Loading Edge Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |

| Value | Name | Description |
|---|---|---|
| 1 | RISING | Rising edge of TIOAx |
| 2 | FALLING | Falling edge of TIOAx |
| 3 | EDGE | Each edge of TIOAx |

**Bit 15 – WAVE** Waveform Mode

| Value | Description |
|---|---|
| 0 | Capture mode is enabled. |
| 1 | Capture mode is disabled (Waveform mode is enabled). |

**Bit 14 – CPCTRG** RC Compare Trigger Enable

| Value | Description |
|---|---|
| 0 | RC Compare has no effect on the counter and its clock. |
| 1 | RC Compare resets the counter and starts the counter clock. |

**Bit 10 – ABETRG** TIOAx or TIOBx External Trigger Selection

| Value | Description |
|---|---|
| 0 | TIOBx is used as an external trigger. |
| 1 | TIOAx is used as an external trigger. |

**Bits 9:8 – ETRGEDG[1:0]** External Trigger Edge Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | The clock is not gated by an external signal. |
| 1 | RISING | Rising edge |
| 2 | FALLING | Falling edge |
| 3 | EDGE | Each edge |

**Bit 7 – LDBDIS** Counter Clock Disable with RB Loading

| Value | Description |
|---|---|
| 0 | Counter clock is not disabled when RB loading occurs. |
| 1 | Counter clock is disabled when RB loading occurs. |

**Bit 6 – LDBSTOP** Counter Clock Stopped with RB Loading

| Value | Description |
|---|---|
| 0 | Counter clock is not stopped when RB loading occurs. |
| 1 | Counter clock is stopped when RB loading occurs. |

**Bits 5:4 – BURST[1:0]** Burst Signal Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | The clock is not gated by an external signal. |
| 1 | XC0 | XC0 is ANDed with the selected clock. |
| 2 | XC1 | XC1 is ANDed with the selected clock. |
| 3 | XC2 | XC2 is ANDed with the selected clock. |

**Bit 3 – CLKI** Clock Invert

| Value | Description |
|---|---|
| 0 | Counter is incremented on rising edge of the clock. |
| 1 | Counter is incremented on falling edge of the clock. |

**Bits 2:0 – TCCLKS[2:0]** Clock Selection
To operate at maximum peripheral clock frequency, refer to “TC Extended Mode Register”.

| Value | Name | Description |
|---|---|---|
| 0 | TIMER_CLOCK1 | Clock selected: internal GCLK[35], GCLK[36] clock signal (from PMC) |
| 1 | TIMER_CLOCK2 | Clock selected: internal System bus clock divided by 8 clock signal (from PMC) |
| 2 | TIMER_CLOCK3 | Clock selected: internal System bus clock divided by 32 clock signal (from PMC) |
| 3 | TIMER_CLOCK4 | Clock selected: internal System bus clock divided by 128 clock signal (from PMC) |
| 4 | TIMER_CLOCK5 | Clock selected: internal slow_clock clock signal (from PMC) |
| 5 | XC0 | Clock selected: XC0 |
| 6 | XC1 | Clock selected: XC1 |
| 7 | XC2 | Clock selected: XC2 |

### 53.7.3 TC Channel Mode Register: Waveform Mode

**Name:** TC_CMRx
**Offset:** 0x04 + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BSWTRG[1:0] | | BEEVT[1:0] | | BCPC[1:0] | | BCPB[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ASWTRG[1:0] | | AEEVT[1:0] | | ACPC[1:0] | | ACPA[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WAVE | WAVSEL[1:0] | | ENETRG | EEVT[1:0] | | EEVTEDG[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CPCDIS | CPCSTOP | BURST[1:0] | | CLKI | TCCLKS[2:0] | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:30 – BSWTRG[1:0]** Software Trigger Effect on TIOBx

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

**Bits 29:28 – BEEVT[1:0]** External Event Effect on TIOBx

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

**Bits 27:26 – BCPC[1:0]** RC Compare Effect on TIOBx

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |

| Value | Name | Description |
|---|---|---|
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

**Bits 25:24 – BCPB[1:0]** RB Compare Effect on TIOBx

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

**Bits 23:22 – ASWTRG[1:0]** Software Trigger Effect on TIOAx

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

**Bits 21:20 – AEEVT[1:0]** External Event Effect on TIOAx

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

**Bits 19:18 – ACPC[1:0]** RC Compare Effect on TIOAx

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

**Bits 17:16 – ACPA[1:0]** RA Compare Effect on TIOAx

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

**Bit 15 – WAVE** Waveform Mode

| Value | Description |
|---|---|
| 0 | Waveform mode is disabled (Capture mode is enabled). |
| 1 | Waveform mode is enabled. |

**Bits 14:13 – WAVSEL[1:0]** Waveform Selection

| Value | Name | Description |
|---|---|---|
| 0 | UP | UP mode without automatic trigger on RC Compare |
| 1 | UPDOWN | UPDOWN mode without automatic trigger on RC Compare |
| 2 | UP_RC | UP mode with automatic trigger on RC Compare |
| 3 | UPDOWN_RC | UPDOWN mode with automatic trigger on RC Compare |

**Bit 12 – ENETRG** External Event Trigger Enable
Whatever the value programmed in ENETRG, the selected external event only controls the TIOAx output and TIOBx if not used as input (trigger event input or other input used).

| Value | Description |
|---|---|
| 0 | The external event has no effect on the counter and its clock. |
| 1 | The external event resets the counter and starts the counter clock. |

**Bits 11:10 – EEVT[1:0]** External Event Selection
Signal selected as external event.

| Value | Name | Description | TIOB Direction |
|---|---|---|---|
| 0 | TIOB | TIOB | Input |
| 1 | XC0 | XC0 | Output |
| 2 | XC1 | XC1 | Output |
| 3 | XC2 | XC2 | Output |

**Note:** If TIOB is chosen as the external event signal, it is configured as an input and no longer generates waveforms and subsequently no IRQs.

**Bits 9:8 – EEVTEDG[1:0]** External Event Edge Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | RISING | Rising edge |
| 2 | FALLING | Falling edge |
| 3 | EDGE | Each edge |

**Bit 7 – CPCDIS** Counter Clock Disable with RC Compare

| Value | Description |
|---|---|
| 0 | Counter clock is not disabled when counter reaches RC. |
| 1 | Counter clock is disabled when counter reaches RC. |

**Bit 6 – CPCSTOP** Counter Clock Stopped with RC Compare

| Value | Description |
|---|---|
| 0 | Counter clock is not stopped when counter reaches RC. |
| 1 | Counter clock is stopped when counter reaches RC. |

**Bits 5:4 – BURST[1:0]** Burst Signal Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | The clock is not gated by an external signal. |
| 1 | XC0 | XC0 is ANDed with the selected clock. |
| 2 | XC1 | XC1 is ANDed with the selected clock. |
| 3 | XC2 | XC2 is ANDed with the selected clock. |

**Bit 3 – CLKI** Clock Invert

| Value | Description |
|---|---|
| 0 | Counter is incremented on rising edge of the clock. |
| 1 | Counter is incremented on falling edge of the clock. |

**Bits 2:0 – TCCLKS[2:0]** Clock Selection
To operate at maximum peripheral clock frequency, refer to “TC Extended Mode Register”.

| Value | Name | Description |
|---|---|---|
| 0 | TIMER_CLOCK1 | Clock selected: internal GCLK[35], GCLK[36] clock signal (from PMC) |
| 1 | TIMER_CLOCK2 | Clock selected: internal System bus clock divided by 8 clock signal (from PMC) |
| 2 | TIMER_CLOCK3 | Clock selected: internal System bus clock divided by 32 clock signal (from PMC) |
| 3 | TIMER_CLOCK4 | Clock selected: internal System bus clock divided by 128 clock signal (from PMC) |
| 4 | TIMER_CLOCK5 | Clock selected: internal slow_clock clock signal (from PMC) |
| 5 | XC0 | Clock selected: XC0 |
| 6 | XC1 | Clock selected: XC1 |
| 7 | XC2 | Clock selected: XC2 |

### 53.7.4 TC Stepper Motor Mode Register

**Name:** TC_SMMRx
**Offset:** 0x08 + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** R/W

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | DOWN | GCEN |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – DOWN** Down Count

| Value | Description |
|---|---|
| 0 | Up counter. |
| 1 | Down counter. |

**Bit 0 – GCEN** Gray Count Enable

| Value | Description |
|---|---|
| 0 | TIOAx [x=0..2] and TIOBx [x=0..2] are driven by internal counter of channel x. |
| 1 | TIOAx [x=0..2] and TIOBx [x=0..2] are driven by a 2-bit Gray counter. |

### 53.7.5 TC Register AB

**Name:** TC_RABx
**Offset:** 0x0C + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RAB[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RAB[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RAB[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RAB[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RAB[31:0]** Register A or Register B
RAB contains the next unread capture Register A or Register B value in real time. It is usually read by the DMA after a request due to a valid load edge on TIOAx.

When DMA is used, the RAB register address must be configured as source address of the transfer.

### 53.7.6 TC Counter Value Register

**Name:** TC_CVx
**Offset:** 0x10 + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CV[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CV[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CV[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CV[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CV[31:0]** Counter Value
CV contains the counter value in real time.

**Important:**

### 53.7.7 TC Register A

**Name:** TC_RAx
**Offset:** 0x14 + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read/Write

This register has access Read-only if TC_CMRx.WAVE = 0, Read/Write if TC_CMRx.WAVE = 1.
This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RA[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RA[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RA[31:0]** Register A
RA contains the Register A value in real time.

**Important:**

### 53.7.8 TC Register B

**Name:** TC_RBx
**Offset:** 0x18 + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read/Write

This register has access Read-only if TC_CMRx.WAVE = 0, Read/Write if TC_CMRx.WAVE = 1.
This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RB[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RB[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RB[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RB[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RB[31:0]** Register B
RB contains the Register B value in real time.

**Important:**

### 53.7.9 TC Register C

**Name:** TC_RCx
**Offset:** 0x1C + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RC[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RC[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RC[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RC[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RC[31:0]** Register C
RC contains the Register C value in real time.

**Important:**

### 53.7.10 TC Interrupt Status Register

**Name:** TC_SRx
**Offset:** 0x20 + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | MTIOB | MTIOA | CLKSTA |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 18 – MTIOB** TIOBx Mirror

| Value | Description |
|---|---|
| 0 | TIOBx is low. If TC_CMRx.WAVE = 0, TIOBx pin is low. If TC_CMRx.WAVE = 1, TIOBx is driven low. |
| 1 | TIOBx is high. If TC_CMRx.WAVE = 0, TIOBx pin is high. If TC_CMRx.WAVE = 1, TIOBx is driven high. |

**Bit 17 – MTIOA** TIOAx Mirror

| Value | Description |
|---|---|
| 0 | TIOAx is low. If TC_CMRx.WAVE = 0, TIOAx pin is low. If TC_CMRx.WAVE = 1, TIOAx is driven low. |
| 1 | TIOAx is high. If TC_CMRx.WAVE = 0, TIOAx pin is high. If TC_CMRx.WAVE = 1, TIOAx is driven high. |

**Bit 16 – CLKSTA** Clock Enabling Status

| Value | Description |
|---|---|
| 0 | Clock is disabled. |
| 1 | Clock is enabled. |

**Bit 7 – ETRGS** External Trigger Status (cleared on read)

| Value | Description |
|---|---|
| 0 | External trigger has not occurred since the last read of the Status Register. |
| 1 | External trigger has occurred since the last read of the Status Register. |

**Bit 6 – LDRBS** RB Loading Status (cleared on read)

| Value | Description |
|---|---|
| 0 | RB Load has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 1. |
| 1 | RB Load has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 0. |

**Bit 5 – LDRAS** RA Loading Status (cleared on read)

| Value | Description |
|---|---|
| 0 | RA Load has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 1. |
| 1 | RA Load has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 0. |

**Bit 4 – CPCS** RC Compare Status (cleared on read)

| Value | Description |
|---|---|
| 0 | RC Compare has not occurred since the last read of the Status Register. |
| 1 | RC Compare has occurred since the last read of the Status Register. |

**Bit 3 – CPBS** RB Compare Status (cleared on read)

| Value | Description |
|---|---|
| 0 | RB Compare has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 0. |
| 1 | RB Compare has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 1. |

**Bit 2 – CPAS** RA Compare Status (cleared on read)

| Value | Description |
|---|---|
| 0 | RA Compare has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 0. |
| 1 | RA Compare has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 1. |

**Bit 1 – LOVRS** Load Overrun Status (cleared on read)

| Value | Description |
|---|---|
| 0 | Load overrun has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 1. |
| 1 | RA or RB have been loaded at least twice without any read of the corresponding register since the last read of the Status Register, if TC_CMRx.WAVE = 0. |

**Bit 0 – COVFS** Counter Overflow Status (cleared on read)

| Value | Description |
|---|---|
| 0 | No counter overflow has occurred since the last read of the Status Register. |
| 1 | A counter overflow has occurred since the last read of the Status Register. |

### 53.7.11 TC Interrupt Enable Register

**Name:** TC_IERx
**Offset:** 0x24 + x*0x40 [x=0..2]
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 7 – ETRGS** External Trigger

**Bit 6 – LDRBS** RB Loading

**Bit 5 – LDRAS** RA Loading

**Bit 4 – CPCS** RC Compare

**Bit 3 – CPBS** RB Compare

**Bit 2 – CPAS** RA Compare

**Bit 1 – LOVRS** Load Overrun

**Bit 0 – COVFS** Counter Overflow

### 53.7.12 TC Interrupt Disable Register

**Name:** TC_IDRx
**Offset:** 0x28 + x*0x40 [x=0..2]
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 7 – ETRGS** External Trigger

**Bit 6 – LDRBS** RB Loading

**Bit 5 – LDRAS** RA Loading

**Bit 4 – CPCS** RC Compare

**Bit 3 – CPBS** RB Compare

**Bit 2 – CPAS** RA Compare

**Bit 1 – LOVRS** Load Overrun

**Bit 0 – COVFS** Counter Overflow

### 53.7.13 TC Interrupt Mask Register

**Name:** TC_IMRx
**Offset:** 0x2C + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – ETRGS** External Trigger

**Bit 6 – LDRBS** RB Loading

**Bit 5 – LDRAS** RA Loading

**Bit 4 – CPCS** RC Compare

**Bit 3 – CPBS** RB Compare

**Bit 2 – CPAS** RA Compare

**Bit 1 – LOVRS** Load Overrun

**Bit 0 – COVFS** Counter Overflow

### 53.7.14 TC Extended Mode Register

**Name:** TC_EMRx
**Offset:** 0x30 + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | NODIVCLK |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TRIGSRCB[1:0] | | | | TRIGSRCA[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bit 8 – NODIVCLK** No Divided Clock

| Value | Description |
|---|---|
| 0 | The selected clock is defined by field TCCLKS in TC_CMRx. |
| 1 | The selected clock is peripheral clock and TCCLKS field (TC_CMRx) has no effect. |

**Bits 5:4 – TRIGSRCB[1:0]** Trigger Source for Input B

| Value | Name | Description |
|---|---|---|
| 0 | EXTERNAL_TIOBx | The trigger/capture input B is driven by external pin TIOBx |
| 1 | PWMx | For TC0 to TC10: The trigger/capture input B is driven internally by the comparator output (see Synchronization with PWM) of the PWMx.<br>For TC11: The trigger/capture input B is driven internally by the GTSUCOMP signal of the Ethernet MAC (GMAC). |

**Bits 1:0 – TRIGSRCA[1:0]** Trigger Source for Input A

| Value | Name | Description |
|---|---|---|
| 0 | EXTERNAL_TIOAx | The trigger/capture input A is driven by external pin TIOAx |
| 1 | PWMx | The trigger/capture input A is driven internally by PWMx |

### 53.7.15 TC Block Control Register

**Name:** TC_BCR
**Offset:** 0xC0
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SYNC |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – SYNC** Synchro Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Asserts the SYNC signal which generates a software trigger simultaneously for each of the channels. |

### 53.7.16 TC Block Mode Register

**Name:** TC_BMR
**Offset:** 0xC4
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | MAXCMP[3:0] | | | | MAXFILT[5:4] | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MAXFILT[3:0] | | | | | AUTOC | IDXPHB | SWAP |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | INVIDX | INVB | INVA | EDGPHA | QDTRANS | SPEEDEN | POSEN | QDEN |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TC2XC2S[1:0] | | TC1XC1S[1:0] | | TC0XC0S[1:0] | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:26 – MAXCMP[3:0]** Maximum Consecutive Missing Pulses

| Value | Description |
|---|---|
| 0 | The flag MPE in TC_QISR never rises. |
| 1–15 | Defines the number of consecutive missing pulses before a flag report. |

**Bits 25:20 – MAXFILT[5:0]** Maximum Filter
Pulses with a period shorter than MAXFILT+1 peripheral clock cycles are discarded. For more details on MAXFILT constraints, see "Input Preprocessing"

| Value | Description |
|---|---|
| 1–63 | Defines the filtering capabilities. |

**Bit 18 – AUTOC** AutoCorrection of missing pulses
0 (DISABLED): The detection and autocorrection function is disabled.

1 (ENABLED): The detection and autocorrection function is enabled.

**Bit 17 – IDXPHB** Index Pin is PHB Pin

| Value | Description |
|---|---|
| 0 | IDX pin of the rotary sensor must drive TIOA1. |

| Value | Description |
|---|---|
| 1 | IDX pin of the rotary sensor must drive TIOB0. |

**Bit 16 – SWAP** Swap PHA and PHB

| Value | Description |
|---|---|
| 0 | No swap between PHA and PHB. |
| 1 | Swap PHA and PHB internally, prior to driving the QDEC. |

**Bit 15 – INVIDX** Inverted Index

| Value | Description |
|---|---|
| 0 | IDX (TIOA1) is directly driving the QDEC. |
| 1 | IDX is inverted before driving the QDEC. |

**Bit 14 – INVB** Inverted PHB

| Value | Description |
|---|---|
| 0 | PHB (TIOB0) is directly driving the QDEC. |
| 1 | PHB is inverted before driving the QDEC. |

**Bit 13 – INVA** Inverted PHA

| Value | Description |
|---|---|
| 0 | PHA (TIOA0) is directly driving the QDEC. |
| 1 | PHA is inverted before driving the QDEC. |

**Bit 12 – EDGPHA** Edge on PHA Count Mode

| Value | Description |
|---|---|
| 0 | Edges are detected on PHA only. |
| 1 | Edges are detected on both PHA and PHB. |

**Bit 11 – QDTRANS** Quadrature Decoding Transparent

| Value | Description |
|---|---|
| 0 | Full quadrature decoding logic is active (direction change detected). |
| 1 | Quadrature decoding logic is inactive (direction change inactive) but input filtering and edge detection are performed. |

**Bit 10 – SPEEDEN** Speed Enabled

| Value | Description |
|---|---|
| 0 | Disabled. |
| 1 | Enables the speed measure on channel 0, the time base being provided by channel 2. |

**Bit 9 – POSEN** Position Enabled

| Value | Description |
|---|---|
| 0 | Disable position. |
| 1 | Enables the position measure on channel 0 and 1. |

**Bit 8 – QDEN** Quadrature Decoder Enabled
Quadrature decoding (direction change) can be disabled using QDTRANS bit.

One of the POSEN or SPEEDEN bits must be also enabled.

| Value | Description |
|---|---|
| 0 | Disabled. |
| 1 | Enables the QDEC (filter, edge detection and quadrature decoding). |

**Bits 5:4 – TC2XC2S[1:0]** External Clock Signal 2 Selection

| Value | Name | Description |
|---|---|---|
| 0 | TCLK2 | Signal connected to XC2: TCLK2 |
| 1 | – | Reserved |
| 2 | TIOA0 | Signal connected to XC2: TIOA0 |
| 3 | TIOA1 | Signal connected to XC2: TIOA1 |

**Bits 3:2 – TC1XC1S[1:0]** External Clock Signal 1 Selection

| Value | Name | Description |
|---|---|---|
| 0 | TCLK1 | Signal connected to XC1: TCLK1 |
| 1 | – | Reserved |
| 2 | TIOA0 | Signal connected to XC1: TIOA0 |
| 3 | TIOA2 | Signal connected to XC1: TIOA2 |

**Bits 1:0 – TC0XC0S[1:0]** External Clock Signal 0 Selection

| Value | Name | Description |
|---|---|---|
| 0 | TCLK0 | Signal connected to XC0: TCLK0 |
| 1 | – | Reserved |
| 2 | TIOA1 | Signal connected to XC0: TIOA1 |
| 3 | TIOA2 | Signal connected to XC0: TIOA2 |

### 53.7.17 TC QDEC Interrupt Enable Register

**Name:** TC_QIER
**Offset:** 0xC8
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MPE | QERR | DIRCHG | IDX |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

**Bit 3 – MPE** Consecutive Missing Pulse Error

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the interrupt when an occurrence of MAXCMP consecutive missing pulses is detected. |

**Bit 2 – QERR** Quadrature Error

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the interrupt when a quadrature error occurs on PHA, PHB. |

**Bit 1 – DIRCHG** Direction Change

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the interrupt when a change on rotation direction is detected. |

**Bit 0 – IDX** Index

| Value | Description |
|---|---|
| 0 | No effect. |

| Value | Description |
|---|---|
| 1 | Enables the interrupt when a rising edge occurs on IDX input. |

### 53.7.18 TC QDEC Interrupt Disable Register

**Name:** TC_QIDR
**Offset:** 0xCC
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MPE | QERR | DIRCHG | IDX |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

**Bit 3 – MPE** Consecutive Missing Pulse Error

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the interrupt when an occurrence of MAXCMP consecutive missing pulses has been detected. |

**Bit 2 – QERR** Quadrature Error

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the interrupt when a quadrature error occurs on PHA, PHB. |

**Bit 1 – DIRCHG** Direction Change

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the interrupt when a change on rotation direction is detected. |

**Bit 0 – IDX** Index

| Value | Description |
|---|---|
| 0 | No effect. |

| Value | Description |
|---|---|
| 1 | Disables the interrupt when a rising edge occurs on IDX input. |

### 53.7.19 TC QDEC Interrupt Mask Register

**Name:** TC_QIMR
**Offset:** 0xD0
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MPE | QERR | DIRCHG | IDX |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 3 – MPE** Consecutive Missing Pulse Error

| Value | Description |
|---|---|
| 0 | The interrupt on the maximum number of consecutive missing pulses specified in MAXCMP is disabled. |
| 1 | The interrupt on the maximum number of consecutive missing pulses specified in MAXCMP is enabled. |

**Bit 2 – QERR** Quadrature Error

| Value | Description |
|---|---|
| 0 | The interrupt on quadrature error is disabled. |
| 1 | The interrupt on quadrature error is enabled. |

**Bit 1 – DIRCHG** Direction Change

| Value | Description |
|---|---|
| 0 | The interrupt on rotation direction change is disabled. |
| 1 | The interrupt on rotation direction change is enabled. |

**Bit 0 – IDX** Index

| Value | Description |
|---|---|
| 0 | The interrupt on IDX input is disabled. |
| 1 | The interrupt on IDX input is enabled. |

### 53.7.20 TC QDEC Interrupt Status Register

**Name:** TC_QISR
**Offset:** 0xD4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DIR |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MPE | QERR | DIRCHG | IDX |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 8 – DIR** Direction
Returns an image of the current rotation direction.

**Bit 3 – MPE** Consecutive Missing Pulse Error

| Value | Description |
|---|---|
| 0 | The number of consecutive missing pulses has not reached the maximum value specified in MAXCMP since the last read of TC_QISR. |
| 1 | An occurrence of MAXCMP consecutive missing pulses has been detected since the last read of TC_QISR. |

**Bit 2 – QERR** Quadrature Error

| Value | Description |
|---|---|
| 0 | No quadrature error since the last read of TC_QISR. |
| 1 | A quadrature error occurred since the last read of TC_QISR. |

**Bit 1 – DIRCHG** Direction Change

| Value | Description |
|---|---|
| 0 | No change on rotation direction since the last read of TC_QISR. |
| 1 | The rotation direction changed since the last read of TC_QISR. |

**Bit 0 – IDX** Index

| Value | Description |
|---|---|
| 0 | No Index input change since the last read of TC_QISR. |
| 1 | The IDX input has changed since the last read of TC_QISR. |

### 53.7.21 TC Fault Mode Register

**Name:** TC_FMR
**Offset:** 0xD8
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ENCF1 | ENCF0 |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – ENCF1** Enable Compare Fault Channel 1

| Value | Description |
|---|---|
| 0 | Disables the FAULT output source (CPCS flag) from channel 1. |
| 1 | Enables the FAULT output source (CPCS flag) from channel 1. |

**Bit 0 – ENCF0** Enable Compare Fault Channel 0

| Value | Description |
|---|---|
| 0 | Disables the FAULT output source (CPCS flag) from channel 0. |
| 1 | Enables the FAULT output source (CPCS flag) from channel 0. |

### 53.7.22 TC Write Protection Mode Register

**Name:** TC_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x544944 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
The Timer Counter clock of the first channel must be enabled to access this register.

See “Register Write Protection” for a list of registers that can be write-protected and Timer Counter clock conditions.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x54494D (“TIM” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x54494D (“TIM” in ASCII). |

# 54. Pulse Density Modulation Interface Controller (PDMIC)

## 54.1 Description

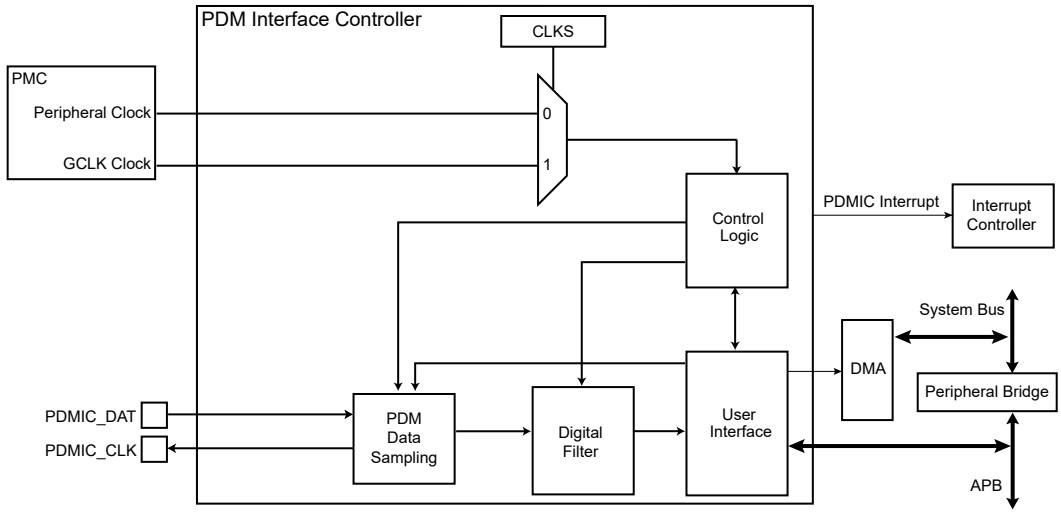
The Pulse Density Modulation Interface Controller (PDMIC) is a PDM interface controller and decoder that support mono PDM format. It integrates a clock generator driving the PDM microphone and embeds filters which decimate the incoming bitstream to obtain most common audio rates.

## 54.2 Embedded Characteristics

- 16-bit Resolution
- DMA Controller Support
- Up to 4 Conversions Stored
- PDM Clock Source can be Independent from Core Clock
- Register Write Protection

## 54.3 Block Diagram

**Figure 54-1. PDMIC Block Diagram**




## 54.4 Signal Description

**Table 54-1. PDMIC Pin Description**

| Pin Name | Description | Type |
|---|---|---|
| PDMIC_CLK | Pulse Density Modulation Bitstream Sampling Clock | Output |
| PDMIC_DAT | Pulse Density Modulation Data | Input |

## 54.5 Product Dependencies

### 54.5.1 I/O Lines

The pins used for interfacing the PDMIC are multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the peripheral functions to PDMIC pins.

### 54.5.2 Power Management

The PDMIC is not continuously clocked. The user must first enable the PDMIC peripheral clocks (PDMIC Channel 0 and PDMIC Channel 1) and the PDMIC Generic Clock in the Power Management Controller (PMC) before using the controller.

### 54.5.3 Interrupt Sources

The PDMIC interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the PDMIC interrupt requires the Interrupt Controller to be programmed first.

## 54.6 Functional Description

### 54.6.1 PDM Interface

#### 54.6.1.1 Description

The PDM clock (PDMIC_CLK) is used to sample the PDM bitstream.

The PDMIC_CLK frequency range is between peripheral clock/2 and peripheral clock/256 or between GCLK clock/2 and GCLK clock/256, depending on the selected clock source.

The GCLK clock frequency must always be at least three times lower than the peripheral clock frequency.

The field PRESCAL in the Mode Register (PDMIC_MR) must be programmed in order to provide a PDMIC_CLK frequency compliant with the microphone parameters.

#### 54.6.1.2 Start-up Sequence

To start processing the bitstream coming from the PDM interface, follow the steps below:

1. Clear all bits in the Control Register (PDMIC_CR) or compute a soft reset using the SWRST bit of PDMIC_CR.
2. Configure the PRESCAL field in PDMIC_MR according to the microphone specifications.
3. Enable the PDM mode and start the conversions using the ENPDM bit in PDMIC_CR.

### 54.6.2 Digital Signal Processing (Digital Filter)

#### 54.6.2.1 Description

The PDMIC includes a DSP section containing a decimation filter, a droop compensation filter, a sixth-order low pass filter, a first-order high pass filter and an offset and gain compensation stage. A block diagram of the DSP section is represented in Figure 54-2.

Data processed by the filtering section are two’s complement signals defined on 24 bits.

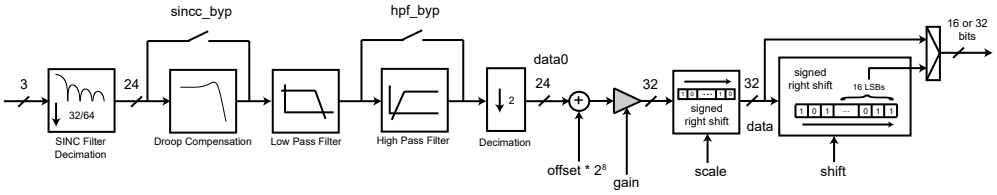
The filtering of the decimation stage is performed by a fourth-order sinc-based filter whose zeros are placed in order to minimize aliasing effects of the decimation. The decimation ratio of this filter is either 32 or 64. The droop induced by this filter can be compensated by the droop compensation stage.

The sixth-order low pass filter is used to decimate the sinc filter output by a ratio of 2.

An optional first-order high pass filter is implemented in order to eliminate the DC component of the incoming signal.

The overall decimation ratio of this DSP section is either 64 or 128. This fits an audio sampling rate of 48 kHz with a PDM microphone sampling frequency of either 3.072 or 6.144 MHz. The frequency response of the filters optimizes the gain flatness between 0 and 20 kHz (when the droop compensation filter is implemented and the high pass filter is bypassed) and highly reduces the aliasing effects of the decimation.

**Figure 54-2. DSP Block Diagram**




### 54.6.2.2 Decimation Filter

The sigma-delta architecture of the PDM microphone implies a filtering and a decimation of the bitstream at the output of the microphone bitstream. The decimation filter decimates the bitstream by either 32 or 64. To perform this operation, a fourth-order sinc filter with an Over-Sampling Ratio (OSR) of 32 or 64 is implemented with the following transfer function:

$$H(z) = \frac{1}{\mathrm{OSR}^4}\left(\sum_{i=0}^{\mathrm{OSR}-1} z^{-i}\right)^4$$

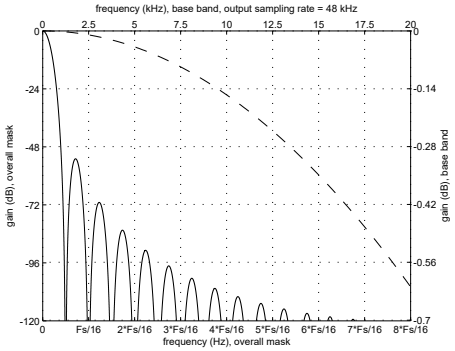
The DC gain of this filter is unity and does not depend on its OSR. However, as it generates a fourth-order zero at Fs/OSR frequency multiples (Fs being the sampling frequency of the microphone), the frequency response of the decimation filter depends on the OSR parameter. See 54.6.2.3 Droop Compensation for frequency plots.

Its non-flat frequency response can be compensated over the 0 to 20 kHz band by using the droop compensation filter when the decimated frequency is set to 48 kHz. See 54.6.2.3 Droop Compensation.

If the decimated sampling rate is modified, the frequency response of this filter is scaled proportionally to the new frequency.

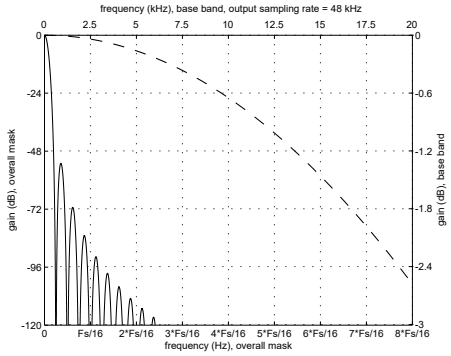
In Figure 54-3 and Figure 54-4, Fs is the sampling rate of the PDM microphone.

**Figure 54-3. Spectral mask of an OSR = 32, Fs = 6.144 MHz, Fourth-Order Sinc Filter: Overall Response (continuous line) and 0 to 20 kHz Bandwidth Response (dashed line)**




The zeros of this filter are located at multiples of Fs/32

**Figure 54-4. Spectral Mask of an OSR = 64, Fs = 3.072 MHz, Fourth-order Sinc Filter: Overall Response (continuous line) and 0 to 20 kHz Bandwidth Response (dashed line)**




The zeros of this filter are located at multiples of Fs/64.

### 54.6.2.3 Droop Compensation

The droop effect introduced by the sinc filter can be compensated in the 0 to 20 kHz by the droop compensation filter (see Figure 54-5). This is a second-order IIR filter which is applied on the signal output by the sinc. The default coefficients of the droop compensation filter are computed to optimize the droop of the sinc filter with the decimated frequency equal to 48 kHz.

This filter compensates the droop of the sinc filter regardless of the OSR value.

If the decimated sampling rate is modified, the frequency response of this filter is scaled proportionally to the new frequency.

This filter can be bypassed by setting the SINBYP bit in the PDMIC DSP Configuration Register 0 (PDMIC_DSPR0).

**Figure 54-5. Droop Compensation Filter Overall Frequency Response**

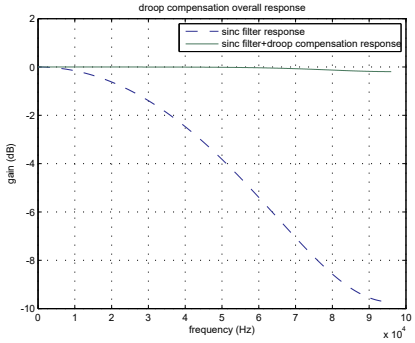
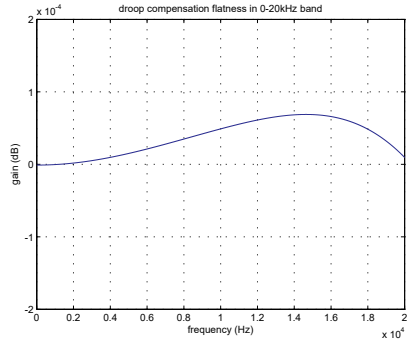



**Figure 54-6. Droop Compensation Filter 0 to 20 kHz Band Flatness**



#### 54.6.2.4 Low Pass Filter

The PDMIC includes a sixth-order IIR filter that performs a low pass transfer function and decimates by 2 the output of the sinc filter. The coefficients are computed for a decimated sampling rate of 48 kHz and optimize the 0 to 20 kHz band flatness while rejecting the aliasing of the PDM microphone by at least 60 dB in the 28 to 48 kHz band.

If the decimated sampling rate is modified, the frequency response of this filter is scaled proportionally to the new frequency.

Figure 54-7 and Figure 54-8 are drawn for an output sampling frequency of 48 kHz.

**Figure 54-7. Low Pass Filter Spectral Mask**

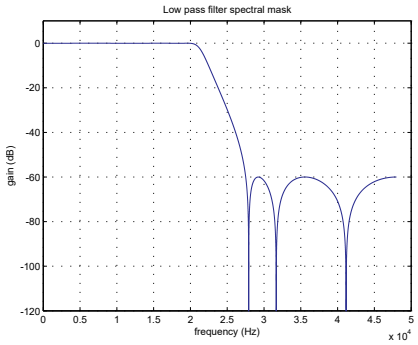
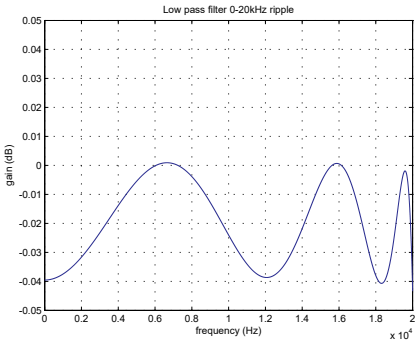



**Figure 54-8. Low Pass Filter Ripple in the 0 to 20 kHz Band**




#### 54.6.2.5 High Pass Filter

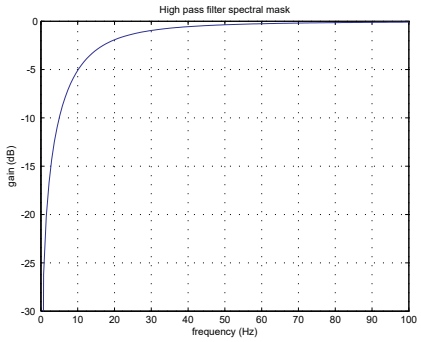
The PDMIC includes an optional first-order IIR filter performing a high pass transfer function after the low pass filter and before the decimation. The coefficients are computed for a decimated sampling rate of 48 kHz to obtain a -3dB cutoff frequency at 15 Hz.

If the decimated sampling rate is modified, the frequency response of this filter is scaled proportionally to the new frequency.

This filter can be bypassed by setting the HPFBYP bit in PDMIC_DSPR0 (see 54.7.8 PDMIC_DSPR0).

Figure 54-9 is drawn for an output sampling frequency of 48 kHz.

**Figure 54-9. High Pass Filter Spectral Mask in the 0 to 100 Hz Band**




#### 54.6.2.6 Gain and Offset Compensation

An offset, a gain, a scaling factor and a shift can be applied to a converted PDM microphone value using the following operation:

$$\text{data} = \frac{\left(\text{data}_0 + \text{offset} \times 2^8\right) \times \text{dgain}}{2^{\text{scale} + \text{shift} + 8}}$$

where:

- $\text{data}_0$ is a signed integer defined on 24 bits. It is the output of the filtering channel.
- offset is a signed integer defined on 16 bits (see PDMIC DSP Configuration Register 1). It is multiplied by $2^8$ to have the same weight as $\text{data}_0$.
- dgain is an unsigned integer defined on 15 bits (see PDMIC DSP Configuration Register 1). Only the 32 MSBs of the multiplication operation are used for scaling and shifting operations. dgain defaults to 0 after reset, which forces CDR to 0. It must be programmed to a non-zero value to read non-zero data into the PDMIC_CDR register.
- scale is an unsigned integer defined on 4 bits (see PDMIC DSP Configuration Register 0). It shifts the multiplication operation result by scale bits to the right. Maximum allowed value is 15.
- shift is an unsigned integer defined on 4 bits (see PDMIC DSP Configuration Register 0). It shifts the multiplication operation result by shift bits to the right. Maximum allowed value is 15.

If the data transfer is configured in 32-bit mode (see PDMIC DSP Configuration Register 0), the $2^{\text{shift}}$ division is not performed and the 32-bit result of the remaining operation is sent.

If the data transfer is configured in 16-bit mode, the $2^{\text{shift}}$ division is performed. The result is then saturated to be within $\pm(2^{15}-1)$ and the 16 LSBs of this saturation operation are sent to the controller as the result of the PDM microphone conversion.

Default parameters are defined to output a 16-bit result whatever the data transfer configuration may be.

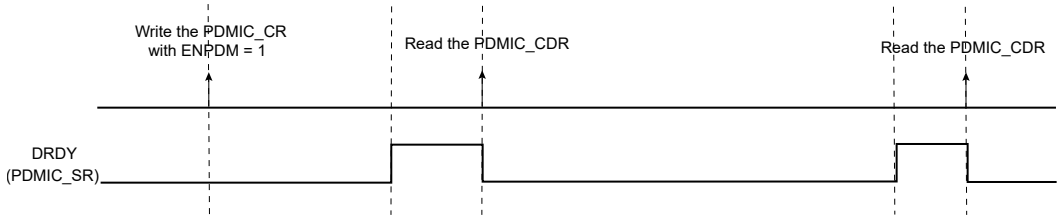
### 54.6.3 Conversion Results

When a conversion is completed, the resulting 16-bit digital value is stored in the PDMIC Converted Data Register (PDMIC_CDR).

The DRDY bit in the Interrupt Status Register (PDMIC_ISR) is set. In the case of a connected DMA Controller channel, DRDY rising triggers a data transfer request. In any case, DRDY can trigger an interrupt.

Reading PDMIC_CDR clears the DRDY flag.

**Figure 54-10. DRDY Flag Behavior**




If PDMIC_CDR is not read before further incoming data is converted, the Overrun Error (OVRE) flag is set in PDMIC_ISR. Likewise, new data converted when DRDY is high sets the OVRE bit (Overrun Error) in PDMIC_ISR. In case of overrun, the newly converted data is lost.

The OVRE flag is automatically cleared when PDMIC_ISR is read.

### 54.6.4 Register Write Protection

To prevent any single software error from corrupting PDMIC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the PDMIC Write Protection Mode Register (PDMIC_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the PDMIC Write Protection Status Register (PDMIC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading PDMIC_WPSR.

The following registers can be write-protected:

- PDMIC Mode Register
- PDMIC DSP Configuration Register 0
- PDMIC DSP Configuration Register 1

## 54.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | PDMIC_CR | 7:0 | | | | ENPDM | | | | SWRST |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | PDMIC_MR | 7:0 | | | | CLKS | | | | |
| | | 15:8 | | PRESCAL[6:0] | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 ... 0x13 | Reserved | | | | | | | | | |
| 0x14 | PDMIC_CDR | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | DATA[23:16] | | | | | | | |
| | | 31:24 | DATA[31:24] | | | | | | | |
| 0x18 | PDMIC_IER | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | OVRE | DRDY |
| 0x1C | PDMIC_IDR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | OVRE | DRDY |
| 0x20 | PDMIC_IMR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | OVRE | DRDY |
| 0x24 | PDMIC_ISR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | FIFOCNT[7:0] | | | | | | | |
| | | 31:24 | | | | | | | OVRE | DRDY |
| 0x28 ... 0x57 | Reserved | | | | | | | | | |
| 0x58 | PDMIC_DSPR0 | 7:0 | | OSR[2:0] | | | SIZE | SINBYP | HPFBYP | |
| | | 15:8 | SHIFT[3:0] | | | | SCALE[3:0] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x5C | PDMIC_DSPR1 | 7:0 | DGAIN[7:0] | | | | | | | |
| | | 15:8 | | DGAIN[14:8] | | | | | | |
| | | 23:16 | OFFSET[7:0] | | | | | | | |
| | | 31:24 | OFFSET[15:8] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x60<br>...<br>0xE3 | Reserved | | | | | | | | | |
| 0xE4 | PDMIC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | PDMIC_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |

### 54.7.1 PDMIC Control Register

**Name:** PDMIC_CR
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | ENPDM | | | | SWRST |
| Access | | | | R/W | | | | R/W |
| Reset | | | | 0 | | | | 0 |

**Bit 4 – ENPDM** Enable PDM

| Value | Description |
|---|---|
| 0 | Disables the PDM and stops the conversions. |
| 1 | Enables the PDM and starts the conversions. |

**Bit 0 – SWRST** Software Reset
Warning: The read value of this bit is always 0.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the PDMIC, simulating a hardware reset. |

### 54.7.2 PDMIC Mode Register

**Name:** PDMIC_MR
**Offset:** 0x04
**Reset:** 0x00F00000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PDMIC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | PRESCAL[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | CLKS | | | | |
| Access | | | | R/W | | | | |
| Reset | | | | 0 | | | | |

**Bits 14:8 – PRESCAL[6:0]** Prescaler Rate Selection
PRESCAL determines the frequency of the PDM bitstream sampling clock (PDMIC_CLK):

$$\mathrm{PRESCAL} = \frac{\mathrm{SELCK}}{2 \times f_{\mathrm{PDMIC_CLK}}} - 1$$

where SELCK is either $f_{\text{peripheral clock}}$ or $f_{\text{GCLK clock}}$ depending on the value of bit CLKS ($f_{\text{peripheral clock}}$ or $f_{\text{GCLK clock}}$ is the clock frequency in Hz).

**Bit 4 – CLKS** Clock Source Selection

| Value | Description |
|---|---|
| 0 | Peripheral clock selected |
| 1 | GCLK clock selected (This clock source can be independent of the processor clock.) |

### 54.7.3 PDMIC Converted Data Register

**Name:** PDMIC_CDR
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATA[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DATA[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DATA[31:0]** Data Converted
The filtered output data is placed into this register at the end of a conversion and remains until it is read.

### 54.7.4 PDMIC Interrupt Enable Register

**Name:** PDMIC_IER
**Offset:** 0x18
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | OVRE | DRDY |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 25 – OVRE** Overrun Error Interrupt Enable

**Bit 24 – DRDY** Data Ready Interrupt Enable

### 54.7.5 PDMIC Interrupt Disable Register

**Name:** PDMIC_IDR
**Offset:** 0x1C
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | OVRE | DRDY |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 25 – OVRE** General Overrun Error Interrupt Disable

**Bit 24 – DRDY** Data Ready Interrupt Disable

### 54.7.6 PDMIC Interrupt Mask Register

**Name:** PDMIC_IMR
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | OVRE | DRDY |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 25 – OVRE** General Overrun Error Interrupt Mask

**Bit 24 – DRDY** Data Ready Interrupt Mask

### 54.7.7 PDMIC Interrupt Status Register

**Name:** PDMIC_ISR
**Offset:** 0x24
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | OVRE | DRDY |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FIFOCNT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 25 – OVRE** Overrun Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No overrun error has occurred since the last read of PDMIC_ISR. |
| 1 | At least one overrun error has occurred since the last read of PDMIC_ISR. |

**Bit 24 – DRDY** Data Ready (cleared by reading PDMIC_CDR)

| Value | Description |
|---|---|
| 0 | No data has been converted since the last read of PDMIC_CDR. |
| 1 | At least one data has been converted and is available in PDMIC_CDR. |

**Bits 23:16 – FIFOCNT[7:0]** FIFO Count
Number of conversions available in the FIFO (not a source of interrupt).

### 54.7.8 PDMIC DSP Configuration Register 0

**Name:** PDMIC_DSPR0
**Offset:** 0x58
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PDMIC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SHIFT[3:0] | | | | SCALE[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | OSR[2:0] | | | SIZE | SINBYP | HPFBYP | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | |

**Bits 15:12 – SHIFT[3:0]** Data Shift
Shifts the scaled result by SHIFT bits to the right.

**Bits 11:8 – SCALE[3:0]** Data Scale
Shifts the multiplication operation result by SCALE bits to the right.

**Bits 6:4 – OSR[2:0]** Global Oversampling Ratio
Values not listed are reserved.

| Value | Name | Description |
|---|---|---|
| 0 | 128 | Global Oversampling ratio is 128 (SINC filter oversampling ratio is 64) |
| 1 | 64 | Global Oversampling ratio is 64 (SINC filter oversampling ratio is 32) |

**Bit 3 – SIZE** Data Size

| Value | Description |
|---|---|
| 0 | Converted data size is 16 bits. |
| 1 | Converted data size is 32 bits. |

**Bit 2 – SINBYP** SINCC Filter Bypass

| Value | Description |
|---|---|
| 0 | Droop compensation filter enabled. |
| 1 | Bypasses the droop compensation filter. |

**Bit 1 – HPFBYP** High-Pass Filter Bypass

| Value | Description |
|---|---|
| 0 | High-pass filter enabled. |
| 1 | Bypasses the high-pass filter. |

### 54.7.9 PDMIC DSP Configuration Register 1

**Name:** PDMIC_DSPR1
**Offset:** 0x5C
**Reset:** 0x00000001
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PDMIC Write Protection Mode Register.

DGAIN and OFFSET values can be determined using the formula in 54.6.2.6 Gain and Offset Compensation.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | OFFSET[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | OFFSET[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | DGAIN[14:8] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DGAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**Bits 31:16 – OFFSET[15:0]** Offset Correction
Offset correction to apply to the final result.

**Bits 14:0 – DGAIN[14:0]** Gain Correction
Gain correction to apply to the final result.

### 54.7.10 PDMIC Write Protection Mode Register

**Name:** PDMIC_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x414443 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit.<br>Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See 54.6.4 Register Write Protection for the list of registers that can be write-protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x414443 (“ADC” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x414443 (“ADC” in ASCII). |

### 54.7.11 PDMIC Write Protection Status Register

**Name:** PDMIC_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of PDMIC_WPSR. |
| 1 | A write protection violation has occurred since the last read of PDMIC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 55. Pulse Width Modulation Controller (PWM)

## 55.1 Description

The Pulse Width Modulation Controller (PWM) generates output pulses on 4 channels independently according to parameters defined per channel. Each channel controls two complementary square output waveforms. Characteristics of the output waveforms such as period, duty-cycle, polarity and dead-times (also called dead-bands or non-overlapping times) are configured through the user interface. Each channel selects and uses one of the clocks provided by the clock generator. The clock generator provides several clocks resulting from the division of the PWM peripheral clock. External triggers can be managed to allow output pulses to be modified in real time.

All accesses to the PWM are made through registers mapped on the peripheral bus. All channels integrate a double buffering system in order to prevent an unexpected output waveform while modifying the period, the spread spectrum, the duty-cycle or the dead-times.

Channels can be linked together as synchronous channels to be able to update their duty-cycle or dead-times at the same time.

The update of duty-cycles of synchronous channels can be performed by the DMA Controller channel which offers buffer transfer without processor Intervention.

The PWM includes a spread-spectrum counter to allow a constantly varying period (only for Channel 0). This counter may be useful to minimize electromagnetic interference or to reduce the acoustic noise of a PWM driven motor.

The PWM provides 1 independent comparison units capable of comparing a programmed value to the counter of the synchronous channels (counter of channel 0). These comparisons are intended to generate software interrupts, to trigger pulses on the 2 independent event lines (in order to synchronize ADC conversions with a lot of flexibility independently of the PWM outputs) and to trigger DMA Controller transfer requests.

PWM outputs can be overridden synchronously or asynchronously to their channel counter.

The PWM provides a fault protection mechanism with 6 fault inputs, capable to detect a fault condition and to override the PWM outputs asynchronously (outputs forced to ‘0’, ‘1’ or Hi-Z).

For safety usage, some configuration registers are write-protected.

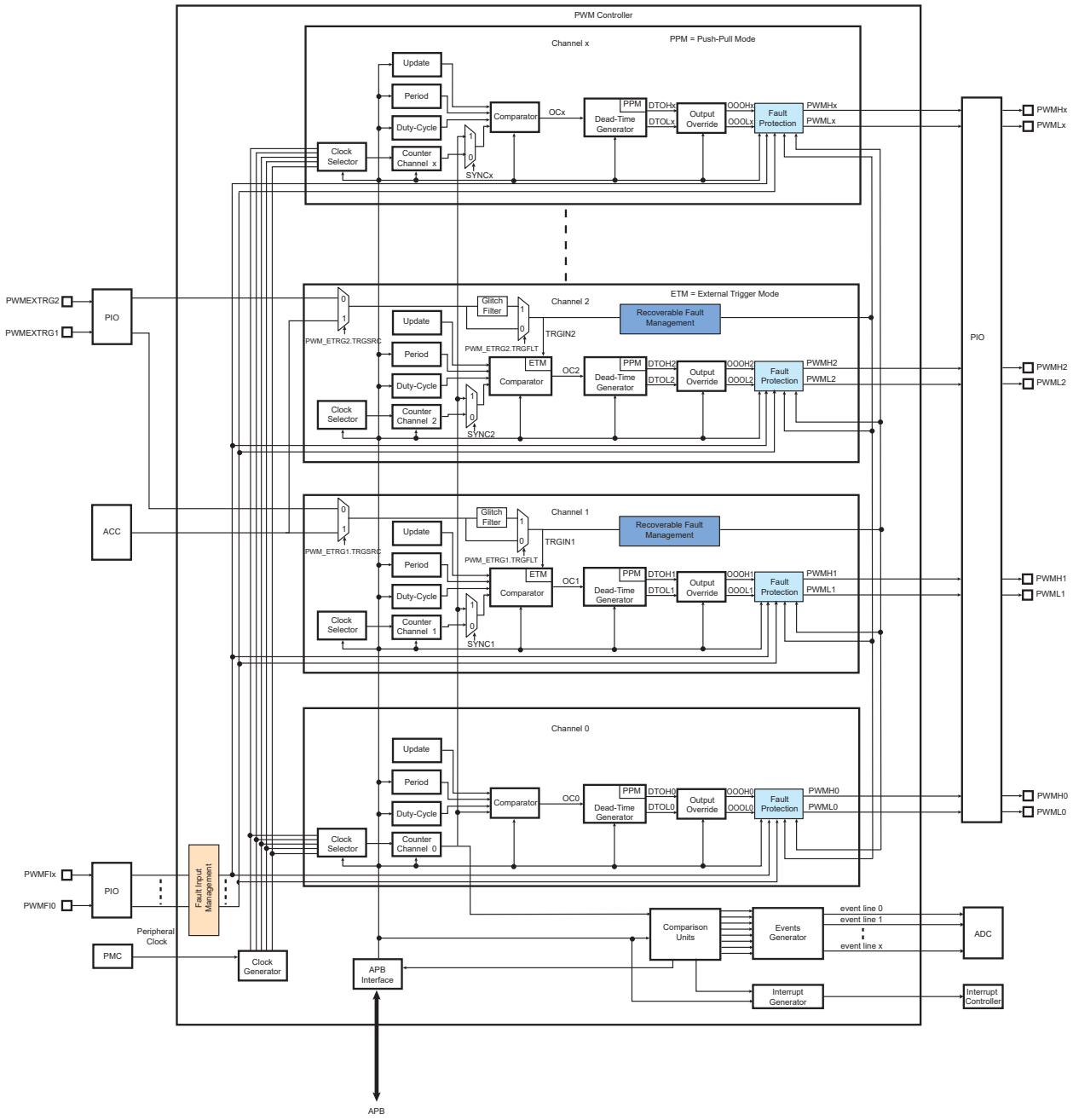
## 55.2 Embedded Characteristics

- 4 Channels
- Common Clock Generator Providing Thirteen Different Clocks
  - A Modulo n Counter Providing Eleven Clocks
  - Two Independent Linear Dividers Working on Modulo n Counter Outputs
- Independent Channels
  - Independent 16-bit Counter for Each Channel
  - Independent Complementary Outputs with 16-bit Dead-Time Generator (Also Called Dead-Band or Non-Overlapping Time) for Each Channel
  - Independent Push-Pull Mode for Each Channel
  - Independent Enable Disable Command for Each Channel

  - Independent Clock Selection for Each Channel
  - Independent Period, Duty-Cycle and Dead-Time for Each Channel
  - Independent Double Buffering of Period, Duty-Cycle and Dead-Times for Each Channel
  - Independent Programmable Selection of The Output Waveform Polarity for Each Channel, with Double Buffering
  - Independent Programmable Center- or Left-aligned Output Waveform for Each Channel
  - Independent Output Override for Each Channel
  - Independent Interrupt for Each Channel, at Each Period for Left-Aligned or Center-Aligned Configuration
  - Independent Update Time Selection of Double Buffering Registers (Polarity, Duty Cycle) for Each Channel, at Each Period for Left-Aligned or Center-Aligned Configuration
- External Trigger Input Management (e.g., for DC/DC or Lighting Control)
  - External PWM Reset Mode
  - External PWM Start Mode
  - Cycle-By-Cycle Duty Cycle Mode
  - Leading-Edge Blanking
- Two 2-bit Gray Up/Down Channels for Stepper Motor Control
- Spread Spectrum Counter to Allow a Constantly Varying Duty Cycle (only for Channel 0)
- Synchronous Channel Mode
  - Synchronous Channels Share the Same Counter
  - Mode to Update the Synchronous Channels Registers after a Programmable Number of Periods
  - Synchronous Channels Supports Connection of one DMA Controller Channel Which Offers Buffer Transfer Without Processor Intervention To Update Duty-Cycle Registers
- 2 Independent Events Lines Intended to Synchronize ADC Conversions
  - Programmable delay for Events Lines to delay ADC measurements
- 1 Comparison Units Intended to Generate Interrupts, Pulses on Event Lines and DMA Controller Transfer Requests
- 6 Programmable Fault Inputs Providing an Asynchronous Protection of PWM Outputs
  - 2 User Driven through PIO Inputs
  - PMC Driven when Crystal Oscillator Clock Fails
  - ADC Controller Driven through Configurable Comparison Function
  - Timer/Counter Driven through Configurable Comparison Function
- Register Write Protection

## 55.3 Block Diagram

**Figure 55-1. Pulse Width Modulation Controller Block Diagram**




**Note:** For a more detailed illustration of the fault protection circuitry, refer to 55.6.2.7 Fault Protection.

## 55.4 I/O Lines Description

Each channel outputs two complementary external I/O lines.

**Table 55-1. I/O Line Description**

| Name | Description | Type |
|---|---|---|
| PWMHx | PWM Waveform Output High for channel x | Output |
| PWMLx | PWM Waveform Output Low for channel x | Output |
| PWMFIx | PWM Fault Input x | Input |
| PWMEXTRGy | PWM Trigger Input y | Input |

## 55.5 Product Dependencies

### 55.5.1 I/O Lines

The pins used for interfacing the PWM are multiplexed with PIO lines. The programmer must first program the PIO controller to assign the desired PWM pins to their peripheral function. If I/O lines of the PWM are not used by the application, they can be used for other purposes by the PIO controller.

All of the PWM outputs may or may not be enabled. If an application requires only four channels, then only four PIO lines are assigned to PWM outputs.

### 55.5.2 Power Management

The PWM is not continuously clocked. The programmer must first enable the PWM clock in the Power Management Controller (PMC) before using the PWM. However, if the application does not require PWM operations, the PWM clock can be stopped when not needed and be restarted later. In this case, the PWM will resume its operations where it left off.

### 55.5.3 Interrupt Sources

The PWM interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the PWM interrupt requires the Interrupt Controller to be programmed first.

### 55.5.4 Fault Inputs

The PWM has the fault inputs connected to the different modules. Refer to the implementation of these modules within the product for detailed information about the fault generation procedure. The PWM receives faults from:

- PIO inputs
- the PMC
- the ADC controller
- Timer/Counters

**Table 55-2. Fault Inputs**

| Fault Generator | External PWM Fault Input Number | Polarity Level(1) | Fault Input ID |
|---|---|---|---|
| PB2 | PWMFI0 | User-defined | 0 |
| PB9 | PWMFI1 | User-defined | 1 |
| PMC | – | To be configured to 1 | 2 |

.....continued

| Fault Generator | External PWM Fault Input Number | Polarity Level(1) | Fault Input ID |
|---|---|---|---|
| ADC | – | To be configured to 1 | 3 |
| Timer0 | – | To be configured to 1 | 4 |
| Timer1 | – | To be configured to 1 | 5 |

**Note:**

1. FPOL field in PWMC_FMR.

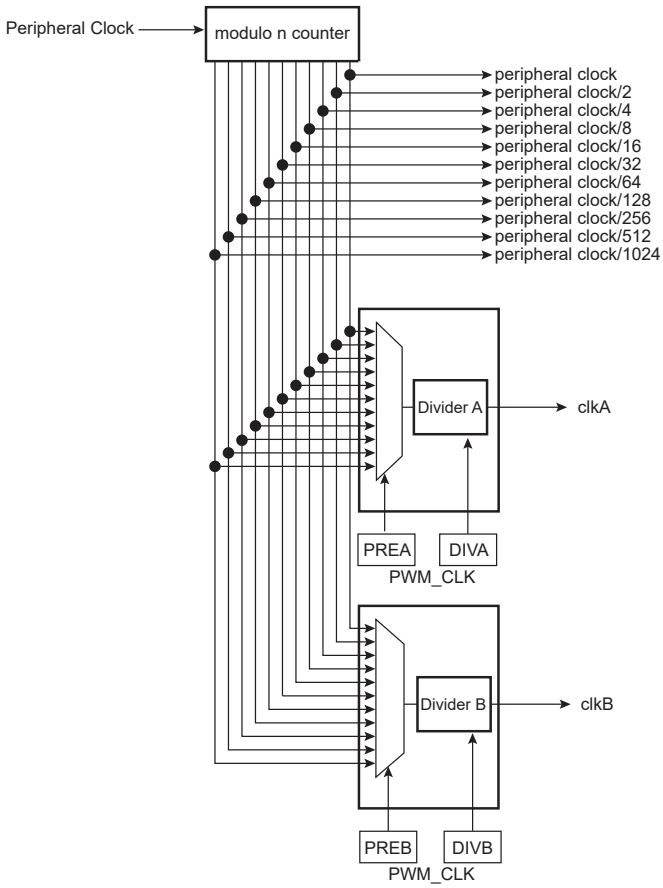
## 55.6 Functional Description

The PWM controller is primarily composed of a clock generator module and 4 channels.

- Clocked by the peripheral clock, the clock generator module provides 13 clocks.
- Each channel can independently choose one of the clock generator outputs.
- Each channel generates an output waveform with attributes that can be defined independently for each channel through the user interface registers.

### 55.6.1 PWM Clock Generator

**Figure 55-2. Functional View of the Clock Generator Block Diagram**




The PWM peripheral clock is divided in the clock generator module to provide different clocks available for all channels. Each channel can independently select one of the divided clocks.

The clock generator is divided into different blocks:

- a modulo n counter which provides 11 clocks:
  - $f_{peripheral\ clock}$
  - $f_{peripheral\ clock}/2$
  - $f_{peripheral\ clock}/4$
  - $f_{peripheral\ clock}/8$
  - $f_{peripheral\ clock}/16$
  - $f_{peripheral\ clock}/32$
  - $f_{peripheral\ clock}/64$
  - $f_{peripheral\ clock}/128$
  - $f_{peripheral\ clock}/256$
  - $f_{peripheral\ clock}/512$
  - $f_{peripheral\ clock}/1024$
- two linear dividers (1, 1/2, 1/3, ... 1/255) that provide two separate clocks: clkA and clkB

Each linear divider can independently divide one of the clocks of the modulo n counter. The selection of the clock to be divided is made according to the PREA (PREB) field of the PWM Clock register (PWM_CLK). The resulting clock clkA (clkB) is the clock selected divided by DIVA (DIVB) field value.

After a reset of the PWM controller, DIVA (DIVB) and PREA (PREB) are set to ‘0’. This implies that after reset clkA (clkB) are turned off.

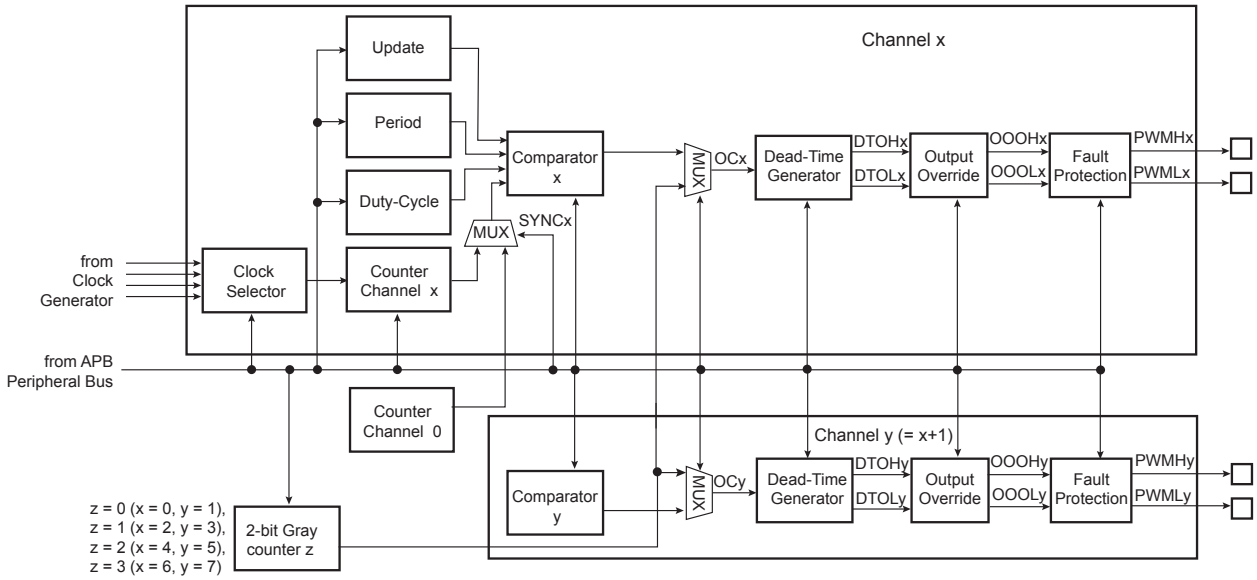
At reset, all clocks provided by the modulo n counter are turned off except the peripheral clock. This situation is also true when the PWM peripheral clock is turned off through the Power Management Controller.

**⚠ CAUTION** Before using the PWM controller, the programmer must first enable the peripheral clock in the Power Management Controller (PMC).

## 55.6.2 PWM Channel

### 55.6.2.1 Channel Block Diagram

**Figure 55-3. Functional View of the Channel Block Diagram**




Each of the 4 channels is composed of six blocks:

- A clock selector which selects one of the clocks provided by the clock generator (described in PWM Clock Generator ).
- A counter clocked by the output of the clock selector. This counter is incremented or decremented according to the channel configuration and comparators matches. The size of the counter is 16 bits.
- A comparator used to compute the OCx output waveform according to the counter value and the configuration. The counter value can be the one of the channel counter or the one of the channel 0 counter according to SYNCx bit in the PWM Sync Channels Mode Register (PWM_SCM).
- A 2-bit configurable Gray counter enables the stepper motor driver. One Gray counter drives 2 channels.
- A dead-time generator providing two complementary outputs (DTOHx/DTOLx) which allows to drive external power control switches safely.
- An output override block that can force the two complementary outputs to a programmed value (OOOHx/OOOLx).
- An asynchronous fault protection mechanism that has the highest priority to override the two complementary outputs (PWMHx/PWMLx) in case of fault detection (outputs forced to ‘0’, ‘1’ or Hi-Z).

### 55.6.2.2 Comparator

The comparator continuously compares its counter value with the channel period defined by CPRD in the PWM Channel Period Register (PWM_CPRDx) and the duty-cycle defined by CDTY in the PWM Channel Duty Cycle Register (PWM_CDTYx) to generate an output signal OCx accordingly.

The different properties of the waveform of the output OCx are:

- the clock selection. The channel counter is clocked by one of the clocks provided by the clock generator described in the previous section. This channel parameter is defined in the CPRE field of the PWM Channel Mode Register (PWM_CMRx). This field is reset at ‘0’.

- the waveform period. This channel parameter is defined in the CPRD field of the PWM_CPRDx register.
  If the waveform is left-aligned, then the output waveform period depends on the counter source clock and can be calculated:
  By using the PWM peripheral clock divided by a given prescaler value “X” (where X = $2^{PREA}$ is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

  $$\frac{(X \times \mathrm{CPRD})}{f_{\text{peripheral clock}}}$$

  By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

  $$\frac{(X \times \mathrm{CPRD} \times \mathrm{DIVA})}{f_{\text{peripheral clock}}} \text{ or } \frac{(X \times \mathrm{CPRD} \times \mathrm{DIVB})}{f_{\text{peripheral clock}}}$$

  If the waveform is center-aligned, then the output waveform period depends on the counter source clock and can be calculated:
  By using the PWM peripheral clock divided by a given prescaler value “X” (where X = $2^{PREA}$ is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

  $$\frac{(2 \times X \times \mathrm{CPRD})}{f_{\text{peripheral clock}}}$$

  By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

  $$\frac{(2 \times X \times \mathrm{CPRD} \times \mathrm{DIVA})}{f_{\text{peripheral clock}}} \text{ or}$$

  $$\frac{(2 \times X \times \mathrm{CPRD} \times \mathrm{DIVB})}{f_{\text{peripheral clock}}}$$
- the waveform duty-cycle. This channel parameter is defined in the CDTY field of the PWM_CDTYx register.
  If the waveform is left-aligned, then:

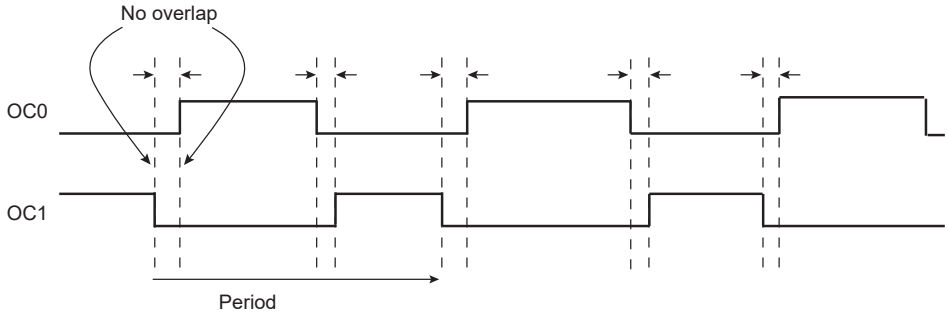
  $$\text{duty cycle} = \frac{(\text{period} - 1/\text{fchannel_x_clock} \times \mathrm{CDTY})}{\text{period}}$$

  If the waveform is center-aligned, then:

  $$\text{duty cycle} = \frac{((\text{period}/2) - 1/\text{fchannel_x_clock} \times \mathrm{CDTY}))}{(\text{period}/2)}$$
- the waveform polarity. At the beginning of the period, the signal can be at high or low level. This property is defined in the CPOL bit of PWM_CMRx. By default, the signal starts by a low level. The DPOLI bit in PWM_CMRx defines the PWM polarity when the channel is disabled (CHIDx = 0 in PWM_SR). For more details, see the figure *Waveform Properties.*
  – DPOLI = 0: PWM polarity when the channel is disabled is the same as the one defined for the beginning of the PWM period.

  - DPOLI = 1: PWM polarity when the channel is disabled is inverted compared to the one defined for the beginning of the PWM period.
- the waveform alignment. The output waveform can be left- or center-aligned. Center-aligned waveforms can be used to generate non-overlapped waveforms. This property is defined in the CALG bit of PWM_CMRx. The default mode is left-aligned.

**Figure 55-4. Non-Overlapped Center-Aligned Waveforms**




**Note:** See the figure Figure 55-5 for a detailed description of center-aligned waveforms.

When center-aligned, the channel counter increases up to CPRD and decreases down to 0. This ends the period.

When left-aligned, the channel counter increases up to CPRD and is reset. This ends the period.

Thus, for the same CPRD value, the period for a center-aligned channel is twice the period for a left-aligned channel.

Waveforms are fixed at 0 when:

- CDTY = CPRD and CPOL = 0 (Note that if TRGMODE = MODE3, the PWM waveform switches to 1 at the external trigger event (see Cycle-By-Cycle Duty Mode)).
- CDTY = 0 and CPOL = 1

Waveforms are fixed at 1 (once the channel is enabled) when:

- CDTY = 0 and CPOL = 0
- CDTY = CPRD and CPOL = 1 (Note that if TRGMODE = MODE3, the PWM waveform switches to 0 at the external trigger event (see Cycle-By-Cycle Duty Mode)).

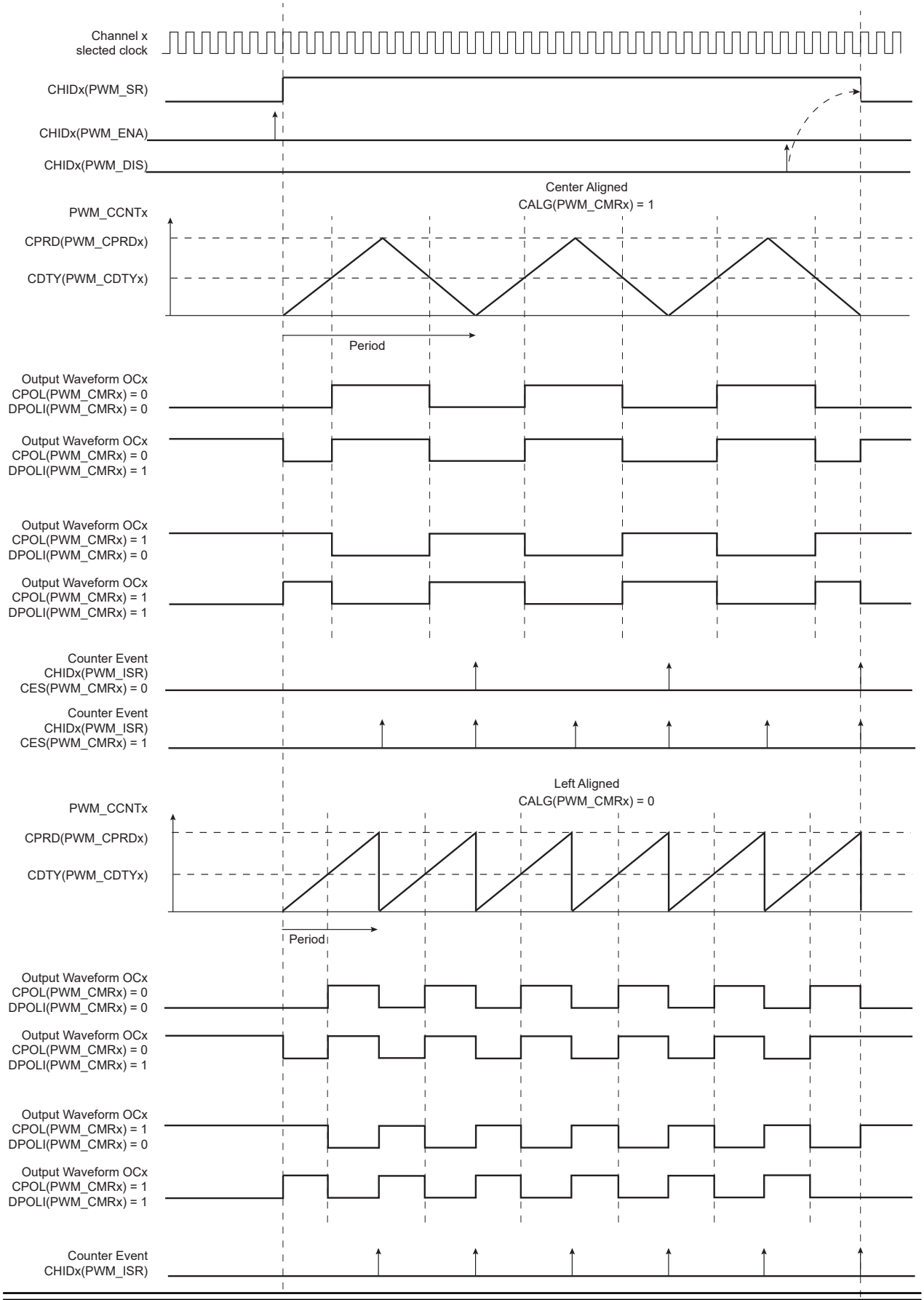
The waveform polarity must be set before enabling the channel. This immediately affects the channel output level.

Modifying CPOL in PWM Channel Mode Register while the channel is enabled can lead to an unexpected behavior of the device being driven by PWM.

In addition to generating the output signals OCx, the comparator generates interrupts depending on the counter value. When the output waveform is left-aligned, the interrupt occurs at the end of the counter period. When the output waveform is center-aligned, the bit CES of PWM_CMRx defines when the channel counter interrupt occurs. If CES is set to ‘0’, the interrupt occurs at the end of the counter period. If CES is set to ‘1’, the interrupt occurs at the end of the counter period and at half of the counter period.

The figure below illustrates the counter interrupts depending on the configuration.

**Figure 55-5. Waveform Properties**



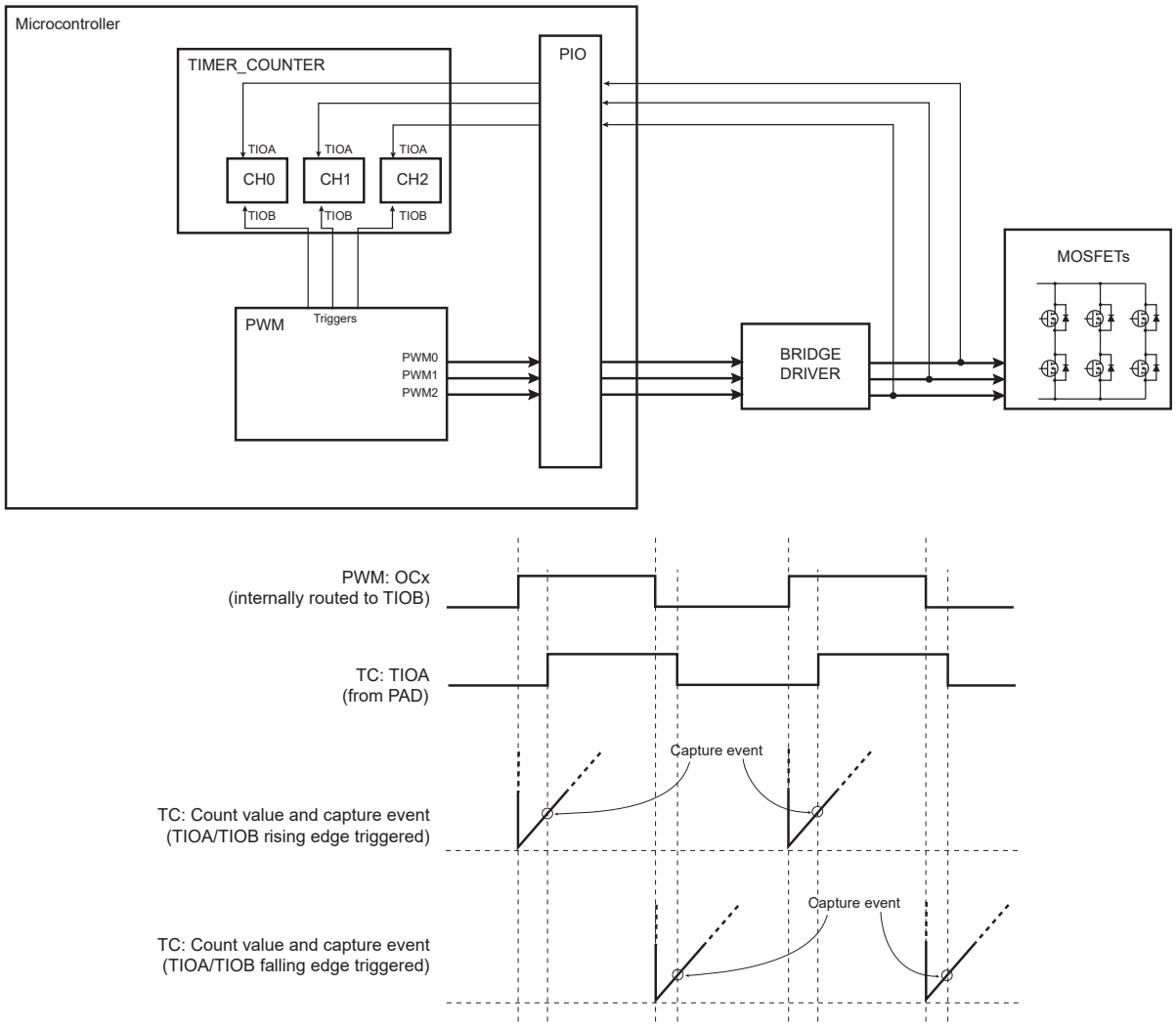
#### 55.6.2.3 Trigger Selection for Timer Counter

The PWM controller can be used as a trigger source for the Timer Counter (TC) to achieve the two application examples described below.

##### 55.6.2.3.1 Delay Measurement

To measure the delay between the channel x comparator output (OCx) and the feedback from the bridge driver of the MOSFETs (see the figure below), the bit TCTS in the PWM Channel Mode Register must be at 0. This defines the comparator output of the channel x as the TC trigger source. The TIOB trigger (TC internal input) is used to start the TC; the TIOA input (from PAD) is used to capture the delay.

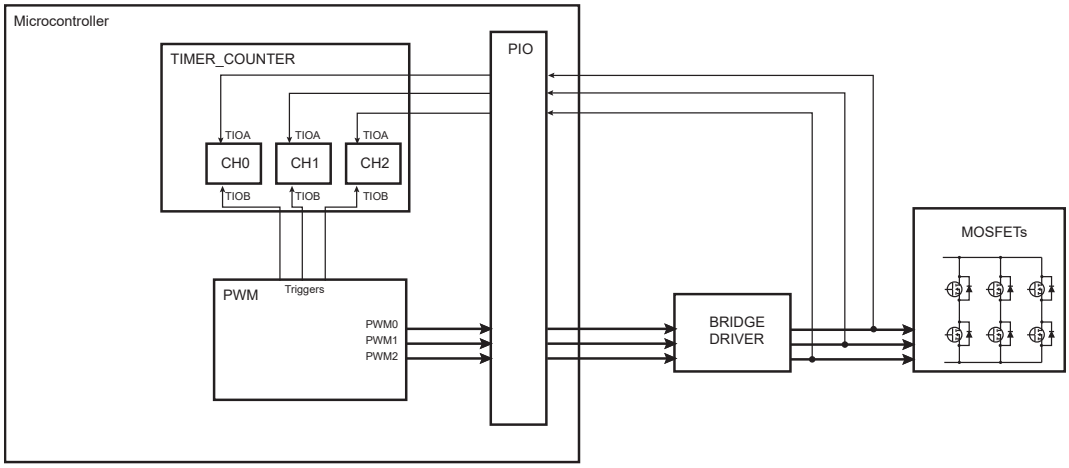
**Figure 55-6. Triggering the TC: Delay Measurement**

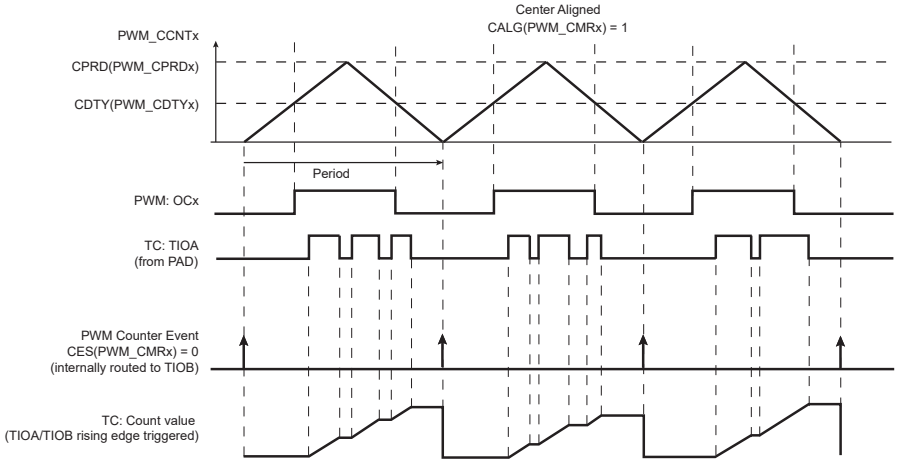


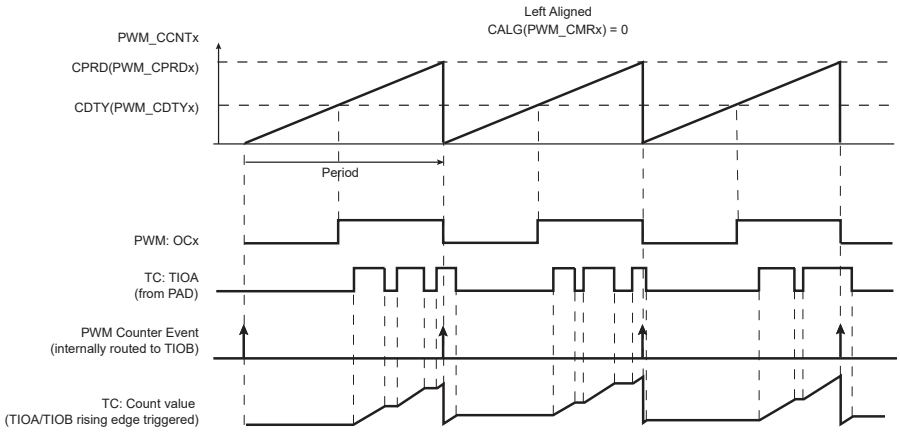

##### 55.6.2.3.2 Cumulated ON Time Measurement

To measure the cumulated “ON” time of MOSFETs (see the figure below), the bit TCTS of the PWM Channel Mode Register must be set to 1 to define the counter event (see the figure *Waveform Properties*) as the Timer Counter trigger source.

**Figure 55-7. Triggering the TC: Cumulated "ON" Time Measurement**



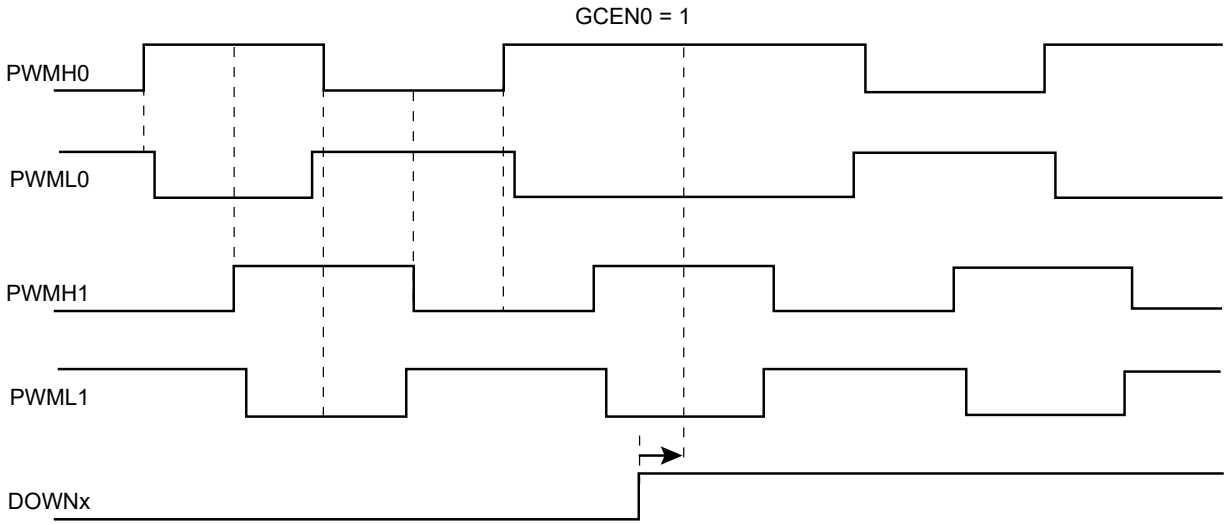

### 55.6.2.4 2-bit Gray Up/Down Counter for Stepper Motor

A pair of channels may provide a 2-bit Gray count waveform on two outputs. Dead-time generator and other downstream logic can be configured on these channels.

Up or Down Count mode can be configured on-the-fly by means of PWM_SMMR configuration registers.

When GCEN0 is set to ‘1’, channels 0 and 1 outputs are driven with a Gray counter.

**Figure 55-8. 2-bit Gray Up/Down Counter**



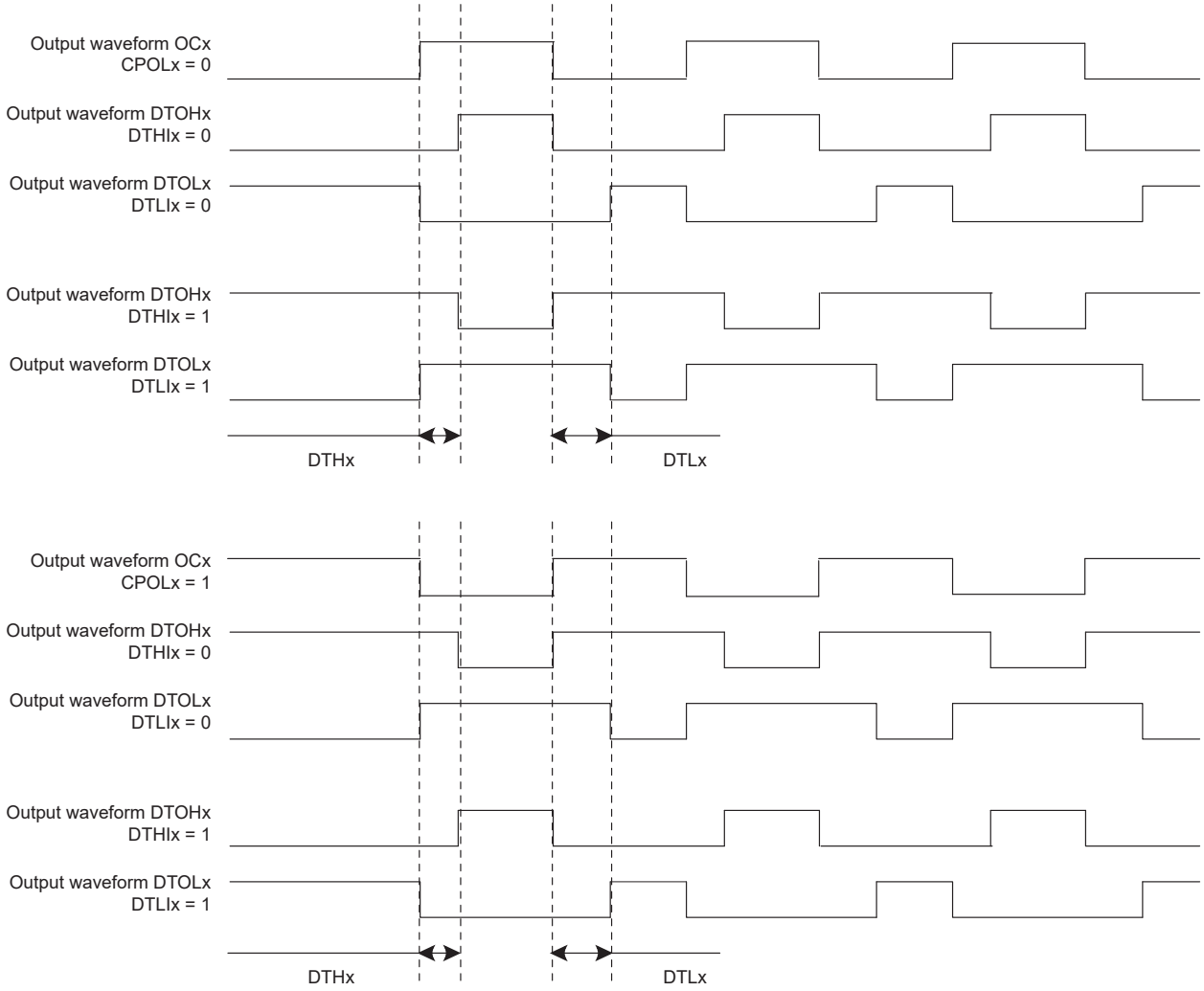

### 55.6.2.5 Dead-Time Generator

The dead-time generator uses the comparator output OCx to provide the two complementary outputs DTOHx and DTOLx, which allows the PWM macrocell to drive external power control switches safely. When the dead-time generator is enabled by setting the bit DTE to 1 or 0 in the PWM Channel Mode Register (PWM_CMRx), dead-times (also called dead-bands or non-overlapping times) are inserted between the edges of the two complementary outputs DTOHx and DTOLx. Note that enabling or disabling the dead-time generator is allowed only if the channel is disabled.

The dead-time is adjustable by the PWM Channel Dead Time Register (PWM_DTx). Each output of the dead-time generator can be adjusted separately by DTH and DTL. The dead-time values can be updated synchronously to the PWM period by using the PWM Channel Dead Time Update Register (PWM_DTUPDx).

The dead-time is based on a specific counter which uses the same selected clock that feeds the channel counter of the comparator. Depending on the edge and the configuration of the dead-time, DTOHx and DTOLx are delayed until the counter has reached the value defined by DTH or DTL. An inverted configuration bit (DTHI and DTLI bit in PWM_CMRx) is provided for each output to invert the dead-time outputs. The following figure shows the waveform of the dead-time generator.

**Figure 55-9. Complementary Output Waveforms**




#### 55.6.2.5.1 PWM Push-Pull Mode

When a PWM channel is configured in Push-Pull mode, the dead-time generator output is managed alternately on each PWM cycle. The polarity of the PWM line during the idle state of the Push-Pull mode is defined by the DPOLI bit in the PWM Channel Mode Register (PWM_CMRx). The Push-Pull mode can be enabled separately on each channel by writing a one to bit PPM in the PWM Channel Mode Register.

**Figure 55-10. PWM Push-Pull Mode**

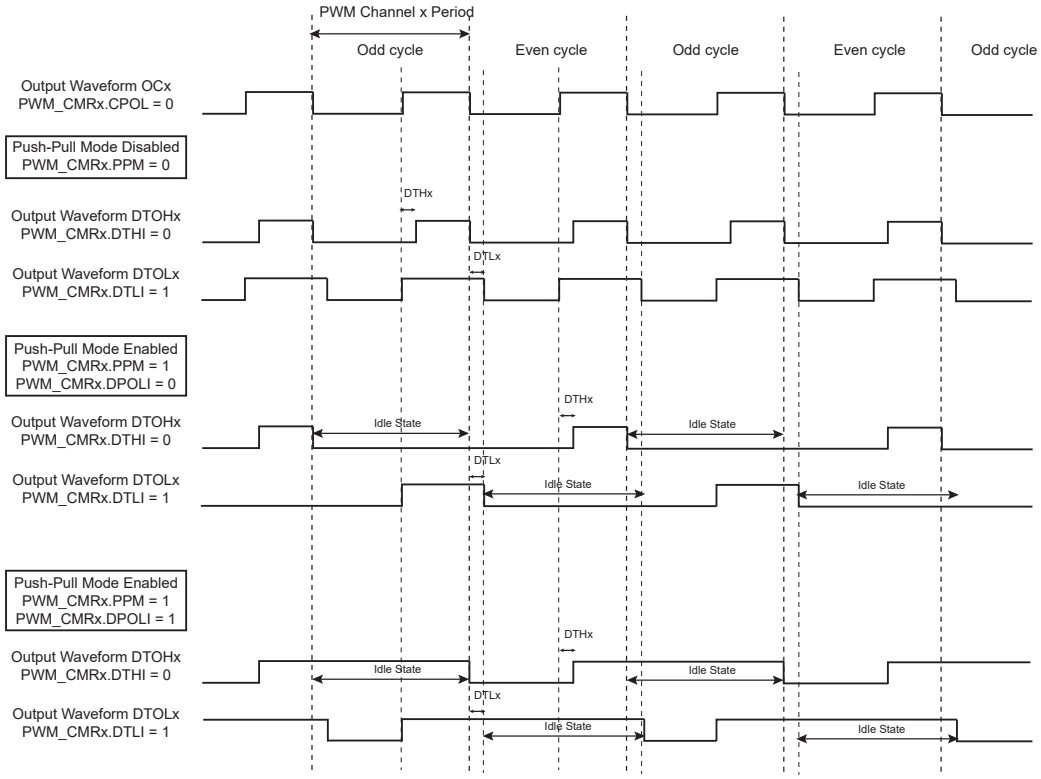


**Figure 55-11. PWM Push-Pull Waveforms: Left-Aligned Mode**

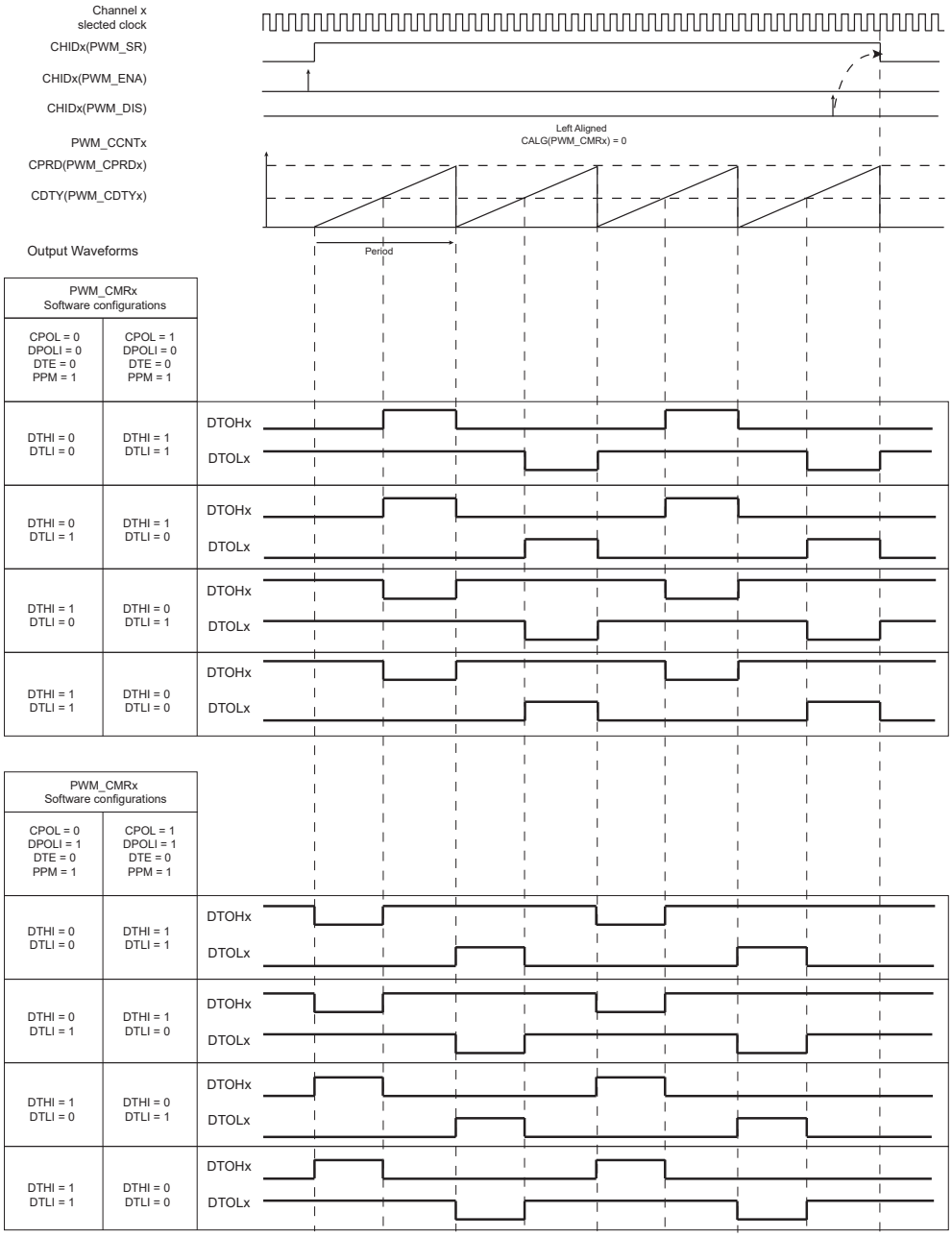
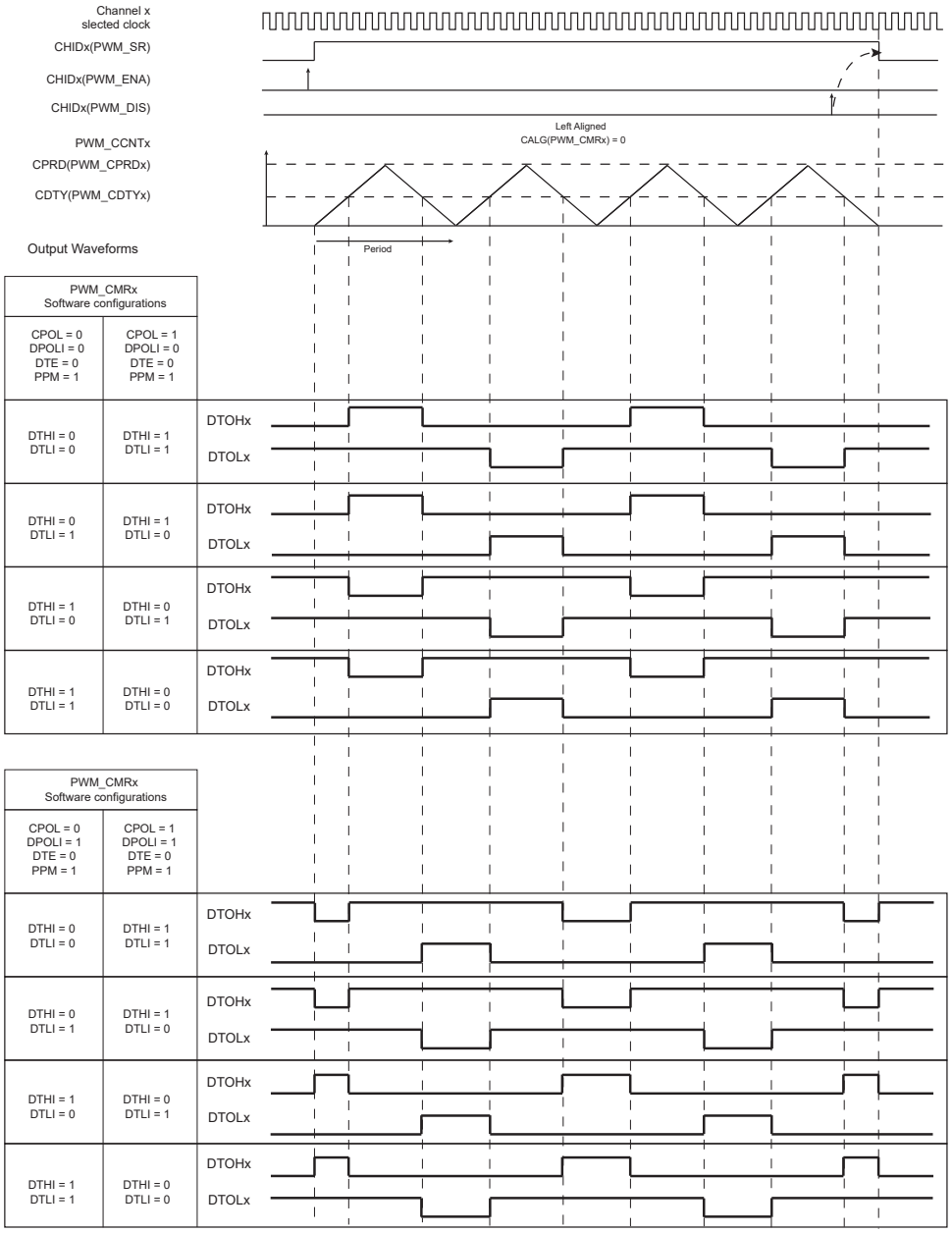


**Figure 55-12. PWM Push-Pull Waveforms: Center-Aligned Mode**




The PWM Push-Pull mode can be useful in transformer-based power converters, such as a half-bridge converter. The Push-Pull mode prevents the transformer core from being saturated by any direct current.

**Figure 55-13. Half-Bridge Converter Application: No Feedback Regulation**

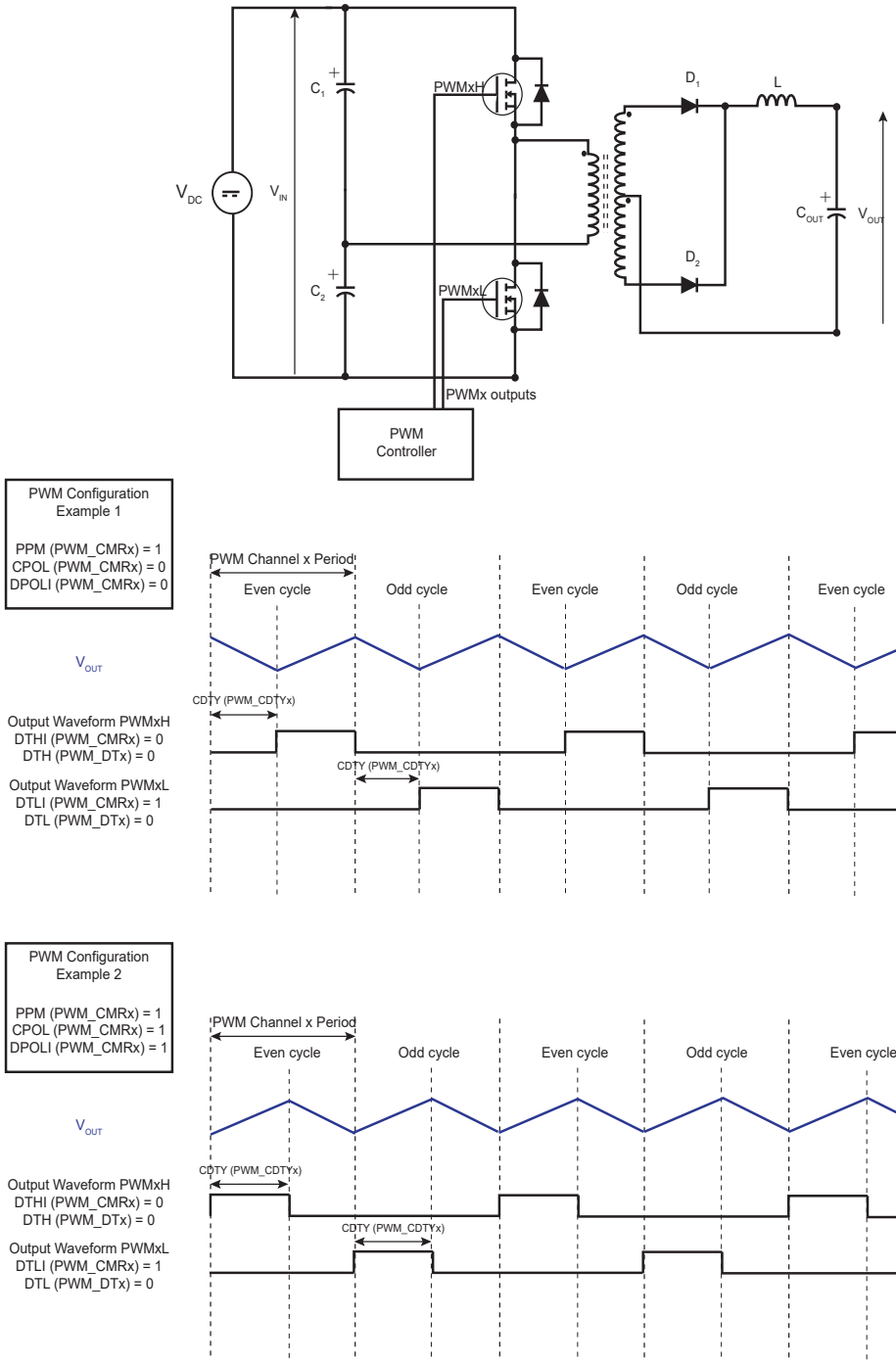
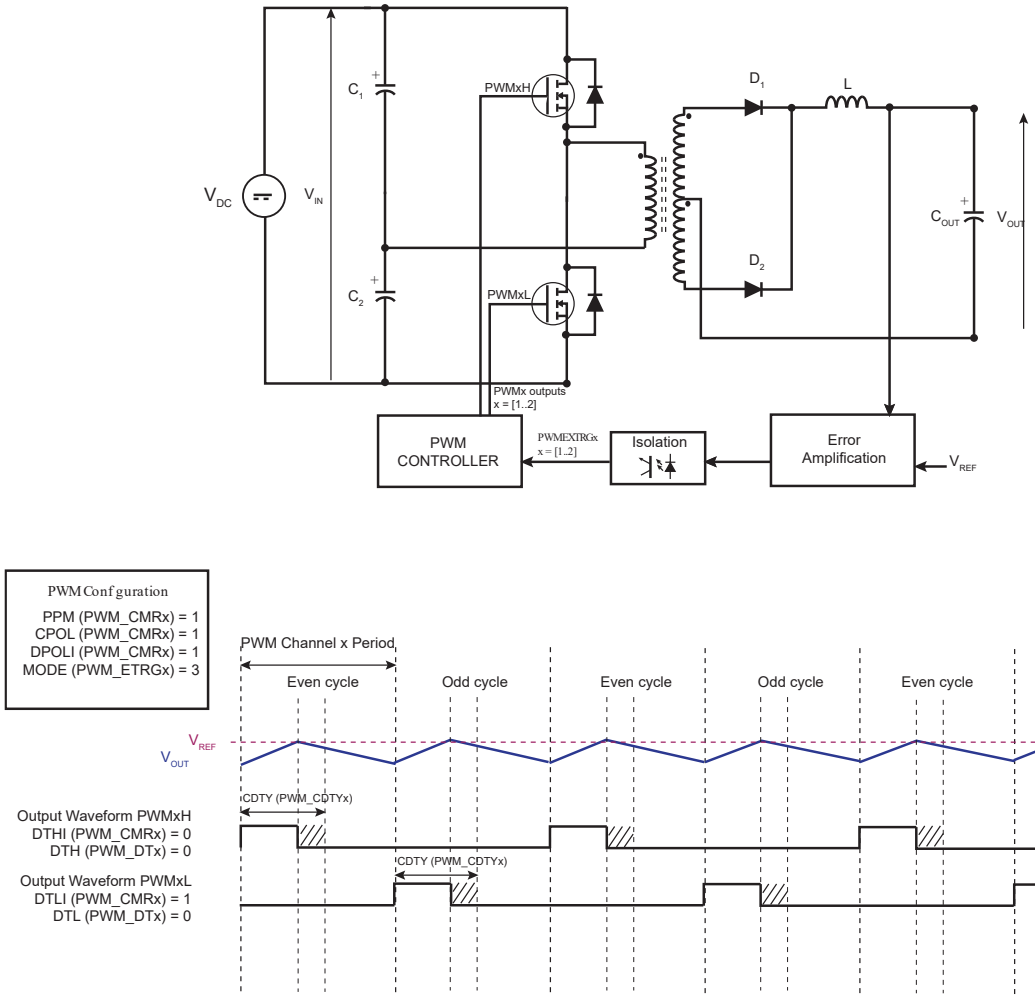


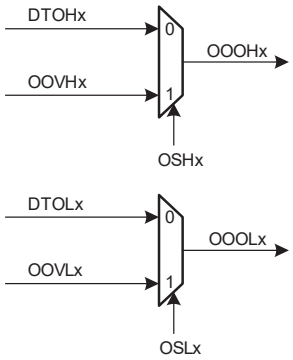
**Figure 55-14. Half-Bridge Converter Application: Feedback Regulation**




### 55.6.2.6 Output Override

The two complementary outputs DTOHx and DTOLx of the dead-time generator can be forced to a value defined by the software.

**Figure 55-15. Override Output Selection**




The fields OSHx and OSLx in the PWM Output Selection Register (PWM_OS) allow the outputs of the dead-time generator DTOHx and DTOLx to be overridden by the value defined in the fields OOVHx and OOVLx in the PWM Output Override Value Register (PWM_OOV).

The set registers PWM Output Selection Set Register (PWM_OSS) and PWM Output Selection Set Update Register (PWM_OSSUPD) enable the override of the outputs of a channel regardless of other channels. In the same way, the clear registers PWM Output Selection Clear Register (PWM_OSC) and PWM Output Selection Clear Update Register (PWM_OSCUPD) disable the override of the outputs of a channel regardless of other channels.

By using buffer registers PWM_OSSUPD and PWM_OSCUPD, the output selection of PWM outputs is done synchronously to the channel counter, at the beginning of the next PWM period.

By using registers PWM_OSS and PWM_OSC, the output selection of PWM outputs is done asynchronously to the channel counter, as soon as the register is written.

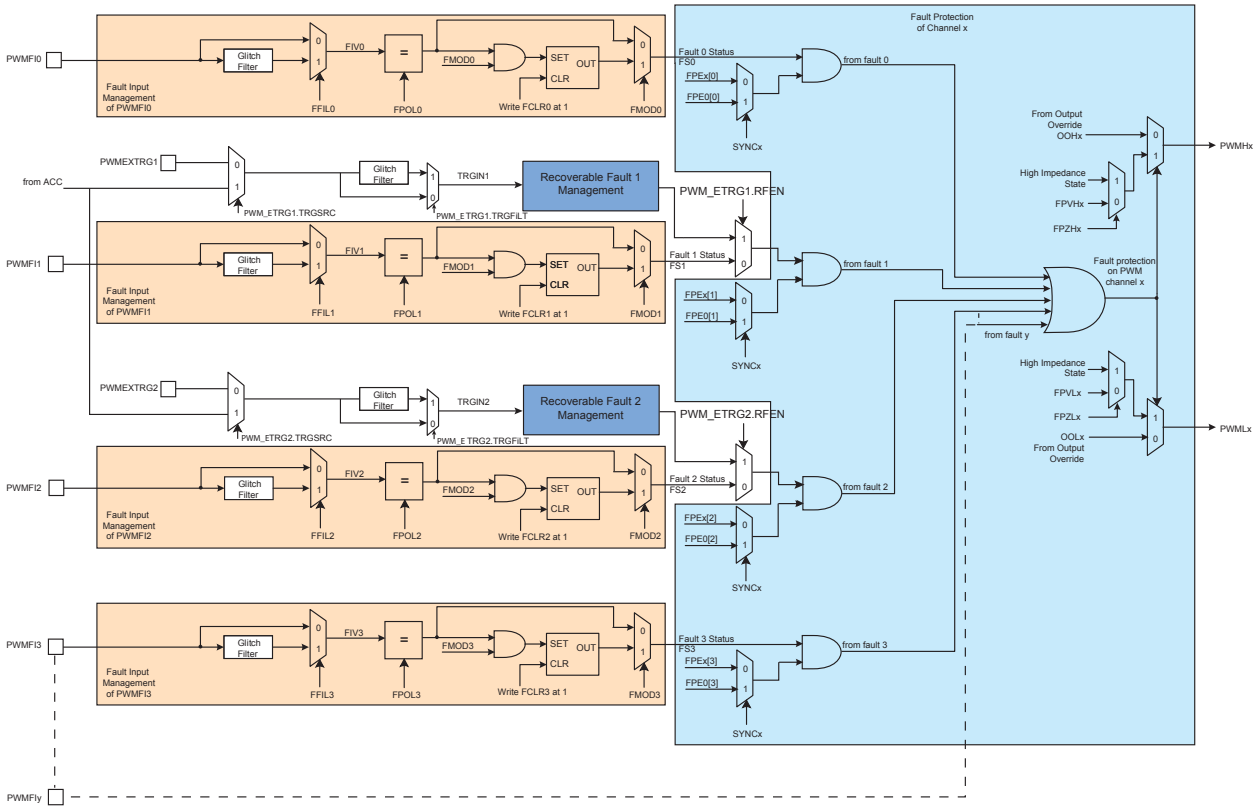
The value of the current output selection can be read in PWM_OS.

While overriding PWM outputs, the channel counters continue to run, only the PWM outputs are forced to user defined values.

### 55.6.2.7 Fault Protection

6 inputs provide fault protection which can force any of the PWM output pairs to a programmable value. This mechanism has priority over output overriding.

**Figure 55-16. Fault Protection**




The polarity level of the fault inputs is configured by the FPOL field in the PWM Fault Mode Register (PWM_FMR). For fault inputs coming from internal peripherals such as ADC or Timer Counter, the polarity level must be FPOL = 1. For fault inputs coming from external GPIO pins the polarity level depends on the user's implementation.

The configuration of the Fault Activation mode (FMOD field in PWMC_FMR) depends on the peripheral generating the fault. If the corresponding peripheral does not have “Fault Clear” management, then the

FMOD configuration to use must be FMOD = 1, to avoid spurious fault detection. Refer to the corresponding peripheral documentation for details on handling fault generation.

Fault inputs may or may not be glitch-filtered depending on the FFIL field in PWM_FMR. When the filter is activated, glitches on fault inputs with a width inferior to the PWM peripheral clock period are rejected.

A fault becomes active as soon as its corresponding fault input has a transition to the programmed polarity level. If the corresponding bit FMOD is set to '0' in PWM_FMR, the fault remains active as long as the fault input is at this polarity level. If the corresponding FMOD field is set to '1', the fault remains active until the fault input is no longer at this polarity level and until it is cleared by writing the corresponding bit FCLR in the PWM Fault Clear Register (PWM_FCR). In the PWM Fault Status Register (PWM_FSR), the field FIV indicates the current level of the fault inputs and the field FIS indicates whether a fault is currently active.

Each fault can be taken into account or not by the fault protection mechanism in each channel. To be taken into account in the channel x, the fault y must be enabled by the bit FPEx[y] in the PWM Fault Protection Enable registers (PWM_FPE1). However, synchronous channels (see Synchronous Channels) do not use their own fault enable bits, but those of the channel 0 (bits FPE0[y]).

The fault protection on a channel is triggered when this channel is enabled and when any one of the faults that are enabled for this channel is active. It can be triggered even if the PWM peripheral clock is not running but only by a fault input that is not glitch-filtered.

When the fault protection is triggered on a channel, the fault protection mechanism resets the counter of this channel and forces the channel outputs to the values defined by the fields FPVHx and FPVLx in the PWM Fault Protection Value Register 1 (PWM_FPV) and fields FPZHx/FPZLx in the PWM Fault Protection Value Register 2, as shown in the table below. The output forcing is made asynchronously to the channel counter.

**Table 55-3. Forcing Values of PWM Outputs by Fault Protection**

| FPZH/Lx | FPVH/Lx | Forcing Value of PWMH/Lx |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | – | High impedance state (Hi-Z) |

**⚠ CAUTION**

- To prevent any unexpected activation of the status flag FSy in PWM_FSR, the FMODy bit can be set to '1' only if the FPOLy bit has been previously configured to its final value.
- To prevent any unexpected activation of the Fault Protection on the channel x, the bit FPEx[y] can be set to '1' only if the FPOLy bit has been previously configured to its final value.

If a comparison unit is enabled (see PWM Comparison Units) and if a fault is triggered in the channel 0, then the comparison cannot match.

As soon as the fault protection is triggered on a channel, an interrupt (different from the interrupt generated at the end of the PWM period) can be generated but only if it is enabled and not masked. The interrupt is reset by reading the interrupt status register, even if the fault which has caused the trigger of the fault protection is kept active.

#### 55.6.2.7.1 Recoverable Fault

The PWM provides a Recoverable Fault mode on fault 1 and 2 (see figure *Fault Protection*).

The recoverable fault signal is an internal signal generated as soon as an external trigger event occurs (see PWM External Trigger Mode).

When the fault 1 or 2 is defined as a recoverable fault, the corresponding fault input pin is ignored and bits FFIL1/2, FMOD1/2 and FFIL1/2 are not taken into account.

The fault 1 is managed as a recoverable fault by the PWMEXTRG1 input trigger when PWM_ETRG1.RFEN = 1, PWM_ENA.CHID1 = 1, and PWM_ETRG1.TRGMODE ≠ 0.

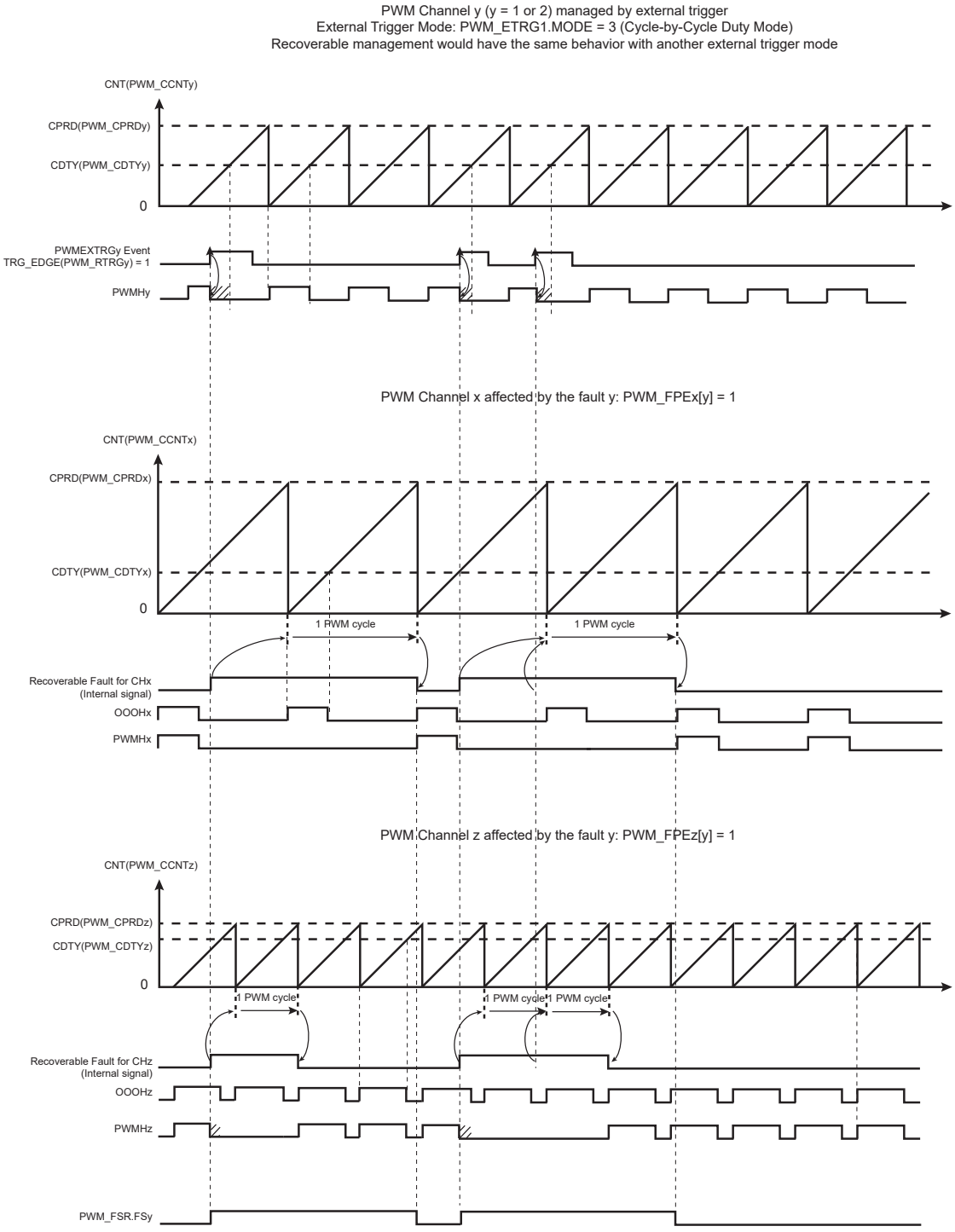
The fault 2 is managed as a recoverable fault by the PWMEXTRG2 input trigger when PWM_ETRG2.RFEN = 1, PWM_ENA.CHID2 = 1, and PWM_ETRG2.TRGMODE ≠ 0.

Recoverable fault 1 and 2 can be taken into account by all channels by enabling the bit FPEx[1/2] in the PWM Fault Protection Enable registers (PWM_FPEx). However the synchronous channels (see Synchronous Channels) do not use their own fault enable bits, but those of the channel 0 (bits FPE0[1/2]).

When a recoverable fault is triggered (according to the PWM_ETRGx.TRGMODE setting), the PWM counter of the affected channels is not cleared (unlike in the classic fault protection mechanism) but the channel outputs are forced to the values defined by the fields FPVHx and FPVLx in the PWM Fault Protection Value Register 1 (PWM_FPV), as per table *Forcing Values of PWM Outputs by Fault Protection*. The output forcing is made asynchronously to the channel counter and lasts from the recoverable fault occurrence to the end of the next PWM cycle (if the recoverable fault is no longer present) (see the figure below).

The recoverable fault does not trigger an interrupt. The Fault Status FSy (with y = 1 or 2) is not reported in the PWM Fault Status Register when the fault y is a recoverable fault.

**Figure 55-17. Recoverable Fault Management**




### 55.6.2.8 Spread Spectrum Counter

The PWM macrocell includes a spread spectrum counter allowing the generation of a constantly varying duty cycle on the output PWM waveform (only for the channel 0). This feature may be useful to minimize electromagnetic interference or to reduce the acoustic noise of a PWM driven motor.

This is achieved by varying the effective period in a range defined by a spread spectrum value which is programmed by the field SPRD in the PWM Spread Spectrum Register (PWM_SSPR). The effective

period of the output waveform is the value of the spread spectrum counter added to the programmed waveform period CPRD in the PWM Channel Period Register (PWM_CPRD0).

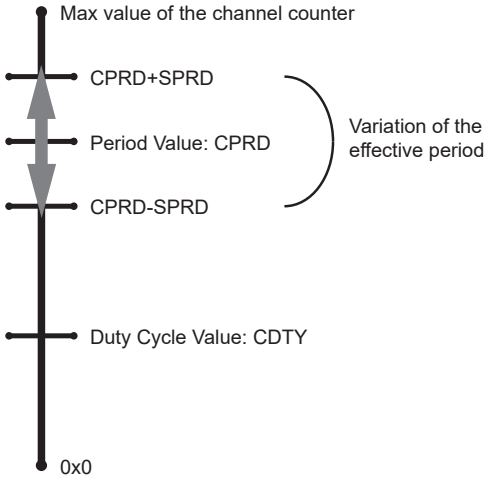
It will cause the effective period to vary from CPRD-SPRD to CPRD+SPRD. This leads to a constantly varying duty cycle on the PWM output waveform because the duty cycle value programmed is unchanged.

The value of the spread spectrum counter can change in two ways depending on the bit SPRDM in PWM_SSPR.

If SPRDM = 0, the Triangular mode is selected. The spread spectrum counter starts to count from -SPRD when the channel 0 is enabled or after reset and counts upwards at each period of the channel counter. When it reaches SPRD, it restarts to count from -SPRD again.

If SPRDM = 1, the Random mode is selected. A new random value is assigned to the spread spectrum counter at each period of the channel counter. This random value is between -SPRD and +SPRD and is uniformly distributed.

**Figure 55-18. Spread Spectrum Counter**




### 55.6.2.9 Synchronous Channels

Some channels can be linked together as synchronous channels. They have the same source clock, the same period, the same alignment and are started together. In this way, their counters are synchronized together.

The synchronous channels are defined by the SYNCx bits in the PWM Sync Channels Mode Register (PWM_SCM). Only one group of synchronous channels is allowed.

When a channel is defined as a synchronous channel, the channel 0 is also automatically defined as a synchronous channel. This is because the channel 0 counter configuration is used by all the synchronous channels.

If a channel x is defined as a synchronous channel, the fields/bits for the channel 0 are used instead of those of channel x:

- CPRE in PWM_CMR0 instead of CPRE in PWM_CMRx (same source clock)
- CPRD in PWM_CPRD0 instead of CPRD in PWM_CPRDx (same period)
- CALG in PWM_CMR0 instead of CALG in PWM_CMRx (same alignment)

Modifying the fields CPRE, CPRD and CALG of for channels with index greater than 0 has no effect on output waveforms.

Because counters of synchronous channels must start at the same time, they are all enabled together by enabling the channel 0 (by the CHID0 bit in PWM_ENA register). In the same way, they are all disabled together by disabling channel 0 (by the CHID0 bit in PWM_DIS register). However, a synchronous channel x different from channel 0 can be enabled or disabled independently from others (by the CHIDx bit in PWM_ENA and PWM_DIS registers).

Defining a channel as a synchronous channel while it is an asynchronous channel (by writing the bit SYNCx to '1' while it was at '0') is allowed only if the channel is disabled at this time (CHIDx = 0 in PWM_SR). In the same way, defining a channel as an asynchronous channel while it is a synchronous channel (by writing the SYNCx bit to '0' while it was '1') is allowed only if the channel is disabled at this time.

The UPDM field (Update Mode) in the PWM_SCM register selects one of the three methods to update the registers of the synchronous channels:

- Method 1 (UPDM = 0): The period value, the duty-cycle values and the dead-time values must be written by the processor in their respective update registers (respectively PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPDx).The update is triggered at the next PWM period as soon as the bit UPDULOCK in the PWM Sync Channels Update Control Register (PWM_SCUC) is set to '1'.
- Method 2 (UPDM = 1): The period value, the duty-cycle values, the dead-time values and the update period value must be written by the processor in their respective update registers (respectively PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPD). The update of the period value and of the dead-time values is triggered at the next PWM period as soon as the bit UPDULOCK in the PWM_SCUC register is set to '1'. The update of the duty-cycle values and the update period value is triggered automatically after an update period defined by the field UPR in the PWM Sync Channels Update Period Register (PWM_SCUP).
- Method 3 (UPDM = 2): Same as Method 2 apart from the fact that the duty-cycle values of ALL synchronous channels are written by the DMA Controller. The user can choose to synchronize the DMA Controller transfer request with a comparison match (see Section 7.3 "PWM Comparison Units"), by the fields PTRM and PTRCS in the PWM_SCM register. The DMA destination address must be configured to access only the PWM DMA Register (PWM_DMAR). The DMA buffer data structure must consist of sequentially repeated duty cycles. The number of duty cycles in each sequence corresponds to the number of synchronized channels. Duty cycles in each sequence must be ordered from the lowest to the highest channel index. The size of the duty cycle is 16 bits.

**Table 55-4. Summary of the Update of Registers of Synchronous Channels**

| Register | UPDM = 0 | UPDM = 1 | UPDM = 2 |
|---|---|---|---|
| Period Value (PWM_CPRDUPDx) | Write by the processor | | |
| | Update is triggered at the next PWM period as soon as the bit UPDULOCK is set to '1' | | |
| Dead-Time Values (PWM_DTUPDx) | Write by the processor | | |
| | Update is triggered at the next PWM period as soon as the bit UPDULOCK is set to '1' | | |

.....continued

| Register | UPDM = 0 | UPDM = 1 | UPDM = 2 |
|---|---|---|---|
| Duty-Cycle Values (PWM_CDTYUPDx) | Write by the processor | Write by the processor | Write by the DMA Controller |
| | Update is triggered at the next PWM period as soon as the bit UPDULOCK is set to '1' | Update is triggered at the next PWM period as soon as the update period counter has reached the value UPR | |
| Update Period Value (PWM_SCUPUPD) | Not applicable | Write by the processor | |
| | Not applicable | Update is triggered at the next PWM period as soon as the update period counter has reached the value UPR | |

#### 55.6.2.9.1 Method 1: Manual write of duty-cycle values and manual trigger of the update

In this mode, the update of the period value, the duty-cycle values and the dead-time values must be done by writing in their respective update registers with the processor (respectively PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPDx).

To trigger the update, the user must use the bit UPDULOCK in the PWM_SCUC register which allows to update synchronously (at the same PWM period) the synchronous channels:

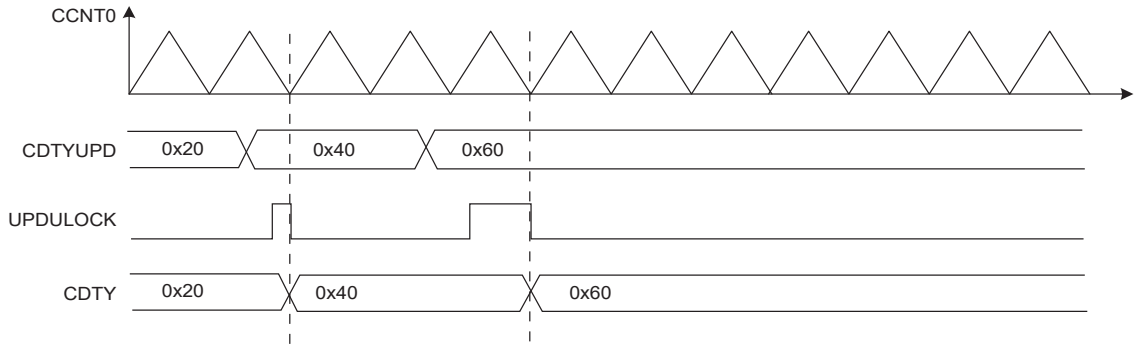
- If the bit UPDULOCK is set to '1', the update is done at the next PWM period of the synchronous channels.
- If the UPDULOCK bit is not set to '1', the update is locked and cannot be performed.

After writing the UPDULOCK bit to '1', it is held at this value until the update occurs, then it is read 0.

Sequence for Method 1:

1. Select the manual write of duty-cycle values and the manual update by setting the UPDM field to '0' in the PWM_SCM register.
2. Define the synchronous channels by the SYNCx bits in the PWM_SCM register.
3. Enable the synchronous channels by writing CHID0 in the PWM_ENA register.
4. If an update of the period value and/or the duty-cycle values and/or the dead-time values is required, write registers that need to be updated (PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPDx).
5. Set UPDULOCK to '1' in PWM_SCUC.
6. The update of the registers will occur at the beginning of the next PWM period. When the UPDULOCK bit is reset, go to Step 4. for new values.

**Figure 55-19. Method 1 (UPDM = 0)**




#### 55.6.2.9.2 Method 2: Manual write of duty-cycle values and automatic trigger of the update

In this mode, the update of the period value, the duty-cycle values, the dead-time values and the update period value must be done by writing in their respective update registers with the processor (respectively PWM_CPRDUPDx, PWM_CDTYUPDx, PWM_DTUPDx and PWM_SCUPUPD).

To trigger the update of the period value and the dead-time values, the user must use the bit UPDULOCK in the PWM_SCUC register, which updates synchronously (at the same PWM period) the synchronous channels:

- If the bit UPDULOCK is set to ‘1’, the update is done at the next PWM period of the synchronous channels.
- If the UPDULOCK bit is not set to ‘1’, the update is locked and cannot be performed.

After writing the UPDULOCK bit to ‘1’, it is held at this value until the update occurs, then it is read 0.

The update of the duty-cycle values and the update period is triggered automatically after an update period.

To configure the automatic update, the user must define a value for the update period by the UPR field in the PWM_SCUP register. The PWM controller waits UPR+1 period of synchronous channels before updating automatically the duty values and the update period value.

The status of the duty-cycle value write is reported in the PWM Interrupt Status Register 2 (PWM_ISR2) by the following flags:

- WRDY: this flag is set to ‘1’ when the PWM Controller is ready to receive new duty-cycle values and a new update period value. It is reset to ‘0’ when the PWM_ISR2 register is read.

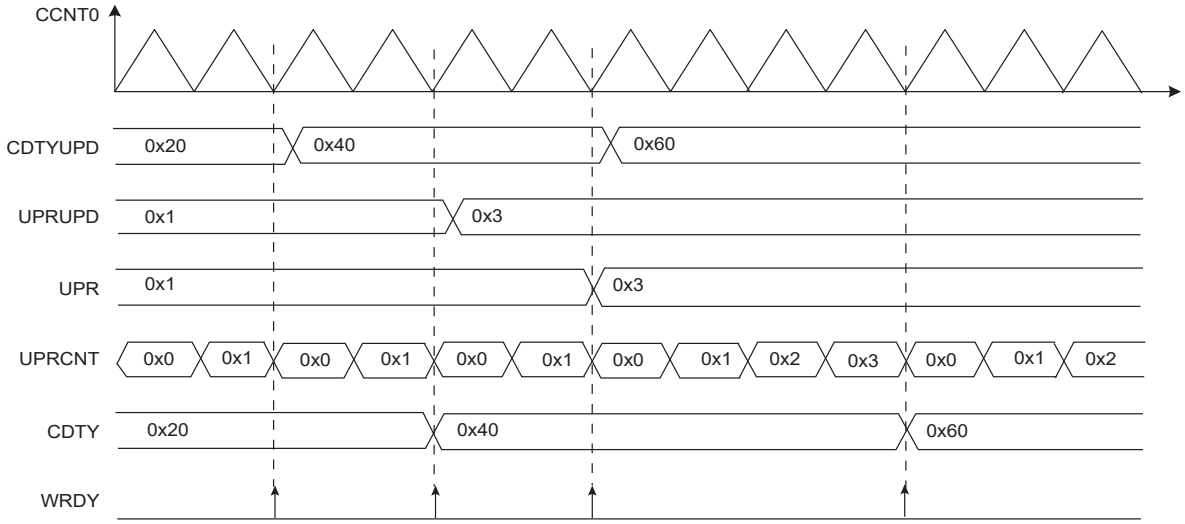
Depending on the interrupt mask in the PWM Interrupt Mask Register 2 (PWM_IMR2), an interrupt can be generated by these flags.

Sequence for Method 2:

1. Select the manual write of duty-cycle values and the automatic update by setting the field UPDM to ‘1’ in the PWM_SCM register
2. Define the synchronous channels by the bits SYNCx in the PWM_SCM register.
3. Define the update period by the field UPR in the PWM_SCUP register.
4. Enable the synchronous channels by writing CHID0 in the PWM_ENA register.
5. If an update of the period value and/or of the dead-time values is required, write registers that need to be updated (PWM_CPRDUPDx, PWM_DTUPDx), else go to Step 8.
6. Set UPDULOCK to ‘1’ in PWM_SCUC.

7. The update of these registers will occur at the beginning of the next PWM period. At this moment the bit UPDULOCK is reset, go to Step 5. for new values.
8. If an update of the duty-cycle values and/or the update period is required, check first that write of new update values is possible by polling the flag WRDY (or by waiting for the corresponding interrupt) in PWM_ISR2.
9. Write registers that need to be updated (PWM_CDTYUPDx, PWM_SCUPUPD).
10. The update of these registers will occur at the next PWM period of the synchronous channels when the Update Period is elapsed. Go to Step 8. for new values.

**Figure 55-20. Method 2 (UPDM = 1)**




#### 55.6.2.9.3 Method 3: Automatic write of duty-cycle values and automatic trigger of the update

In this mode, the update of the duty cycle values is made automatically by the DMA Controller. The update of the period value, the dead-time values and the update period value must be done by writing in their respective update registers with the processor (respectively PWM_CPRDUPDx, PWM_DTUPDx and PWM_SCUPUPD).

To trigger the update of the period value and the dead-time values, the user must use the bit UPDULOCK which allows to update synchronously (at the same PWM period) the synchronous channels:

- If the bit UPDULOCK is set to ‘1’, the update is done at the next PWM period of the synchronous channels.
- If the UPDULOCK bit is not set to ‘1’, the update is locked and cannot be performed.

After writing the UPDULOCK bit to ‘1’, it is held at this value until the update occurs, then it is read 0.

The update of the duty-cycle values and the update period value is triggered automatically after an update period.

To configure the automatic update, the user must define a value for the Update Period by the field UPR in the PWM_SCUP register. The PWM controller waits UPR+1 periods of synchronous channels before updating automatically the duty values and the update period value.

Using the DMA Controller removes processor overhead by reducing its intervention during the transfer. This significantly reduces the number of clock cycles required for a data transfer, which improves microcontroller performance.

The DMA Controller must write the duty-cycle values in the synchronous channels index order. For example if the channels 0, 1 and 3 are synchronous channels, the DMA Controller must write the duty-

cycle of the channel 0 first, then the duty-cycle of the channel 1, and finally the duty-cycle of the channel 3.

The status of the DMA Controller transfer is reported in PWM_ISR2 by the following flags:

- WRDY: this flag is set to ‘1’ when the PWM Controller is ready to receive new duty-cycle values and a new update period value. It is reset to ‘0’ when PWM_ISR2 is read. The user can choose to synchronize the WRDY flag and the DMA Controller transfer request with a comparison match (see PWM Comparison Units), by the fields PTRM and PTRCS in the PWM_SCM register.
- UNRE: this flag is set to ‘1’ when the update period defined by the UPR field has elapsed while the whole data has not been written by the Peripheral DMA Controller. It is reset to ‘0’ when PWM_ISR2 is read.

Depending on the interrupt mask in PWM_IMR2, an interrupt can be generated by these flags.

Sequence for Method 3:

1. Select the automatic write of duty-cycle values and automatic update by setting the field UPDM to 2 in the PWM_SCM register.
2. Define the synchronous channels by the bits SYNCx in the PWM_SCM register.
3. Define the update period by the field UPR in the PWM_SCUP register.
4. Define when the WRDY flag and the corresponding DMA Controller transfer request must be set in the update period by the PTRM bit and the PTRCS field in the PWM_SCM register (at the end of the update period or when a comparison matches).
5. Define the DMA Controller transfer settings for the duty-cycle values and enable it in the DMA Controller registers
6. Enable the synchronous channels by writing CHID0 in the PWM_ENA register.
7. If an update of the period value and/or of the dead-time values is required, write registers that need to be updated (PWM_CPRDUPDx, PWM_DTUPDx), else go to Step 10.
8. Set UPDULOCK to ‘1’ in PWM_SCUC.
9. The update of these registers will occur at the beginning of the next PWM period. At this moment the bit UPDULOCK is reset, go to Step 7. for new values.
10. If an update of the update period value is required, check first that write of a new update value is possible by polling the flag WRDY (or by waiting for the corresponding interrupt) in PWM_ISR2, else go to Step 14.
11. Write the register that needs to be updated (PWM_SCUPUPD).
12. The update of this register will occur at the next PWM period of the synchronous channels when the Update Period is elapsed. Go to Step 10. for new values.
13. Wait for the DMA status flag indicating that the buffer transfer is complete. If the transfer has ended, define a new DMA transfer for new duty-cycle values. Go to Step 5.

14. **Figure 55-21. Method 3 (UPDM = 2 and PTRM = 0)**

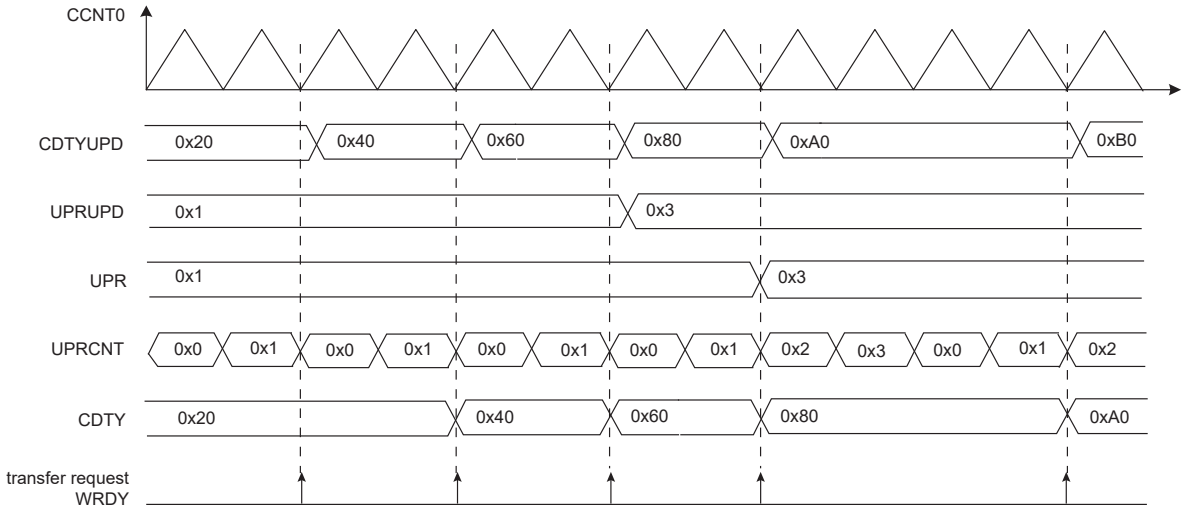
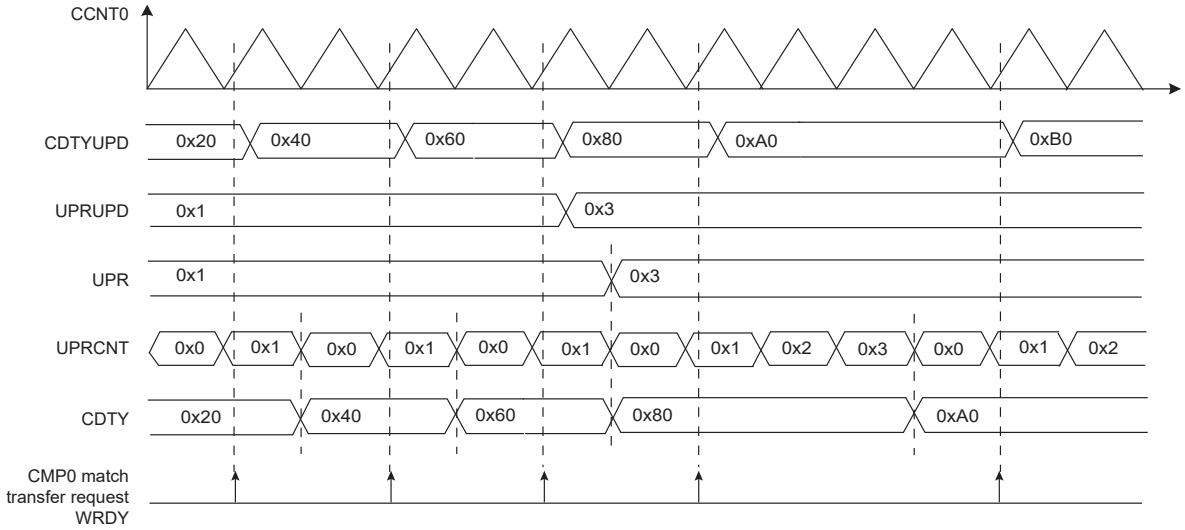



**Figure 55-22. Method 3 (UPDM = 2 and PTRM = 1 and PTRCS = 0)**




### 55.6.2.10 Update Time for Double-Buffering Registers

All channels integrate a double-buffering system in order to prevent an unexpected output waveform while modifying the period, the spread spectrum value, the polarity, the duty-cycle, the dead-times, the output override, and the synchronous channels update period.

This double-buffering system comprises the following update registers:

- PWM Sync Channels Update Period Update Register
- PWM Output Selection Set Update Register
- PWM Output Selection Clear Update Register
- PWM Spread Spectrum Update Register
- PWM Channel Duty Cycle Update Register
- PWM Channel Period Update Register
- PWM Channel Dead Time Update Register
- PWM Channel Mode Update Register

When one of these update registers is written to, the write is stored, but the values are updated only at the next PWM period border. In Left-aligned mode (CALG = 0), the update occurs when the channel counter reaches the period value CPRD. In Center-aligned mode, the update occurs when the channel counter value is decremented and reaches the 0 value.

In Center-aligned mode, it is possible to trigger the update of the polarity and the duty-cycle at the next half period border. This mode concerns the following update registers:

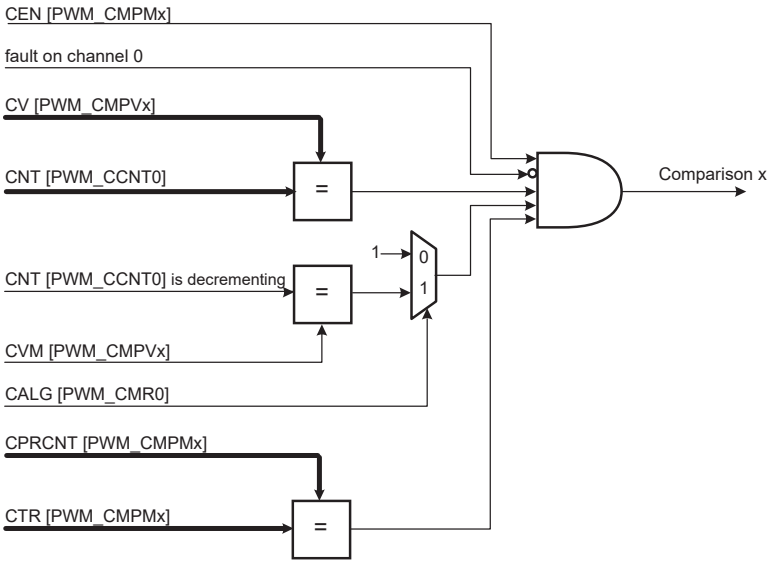
- PWM Channel Duty Cycle Update Register
- PWM Channel Mode Update Register

The update occurs at the first half period following the write of the update register (either when the channel counter value is incrementing and reaches the period value CPRD, or when the channel counter value is decrementing and reaches the 0 value). To activate this mode, the user must write a one to the bit UPDS in the PWM Channel Mode Register.

### 55.6.3 PWM Comparison Units

The PWM provides 1 independent comparison units able to compare a programmed value with the current value of the channel 0 counter (which is the channel counter of all synchronous channels, "Synchronous Channels"). These comparisons are intended to generate pulses on the event lines (used to synchronize ADC, see PWM Event Lines), to generate software interrupts and to trigger DMA Controller transfer requests for the synchronous channels (see Method 3: Automatic write of duty-cycle values and automatic trigger of the update).

**Figure 55-23. Comparison Unit Block Diagram**




The comparison x matches when it is enabled by the bit CEN in the PWM Comparison x Mode Register (PWM_CMPMx for the comparison x) and when the counter of the channel 0 reaches the comparison value defined by the field CV in PWM Comparison x Value Register (PWM_CMPVx for the comparison x). If the counter of the channel 0 is center-aligned (CALG = 1 in PWM Channel Mode Register), the bit CVM in PWM_CMPVx defines if the comparison is made when the counter is counting up or counting down (in Left-alignment mode CALG = 0, this bit is useless).

If a fault is active on the channel 0, the comparison is disabled and cannot match (see Fault Protection).

The user can define the periodicity of the comparison x by the fields CTR and CPR in PWM_CMPMx. The comparison is performed periodically once every CPR+1 periods of the counter of the channel 0, when

the value of the comparison period counter CPRCNT in PWM_CMPMx reaches the value defined by CTR. CPR is the maximum value of the comparison period counter CPRCNT. If CPR = CTR = 0, the comparison is performed at each period of the counter of the channel 0.

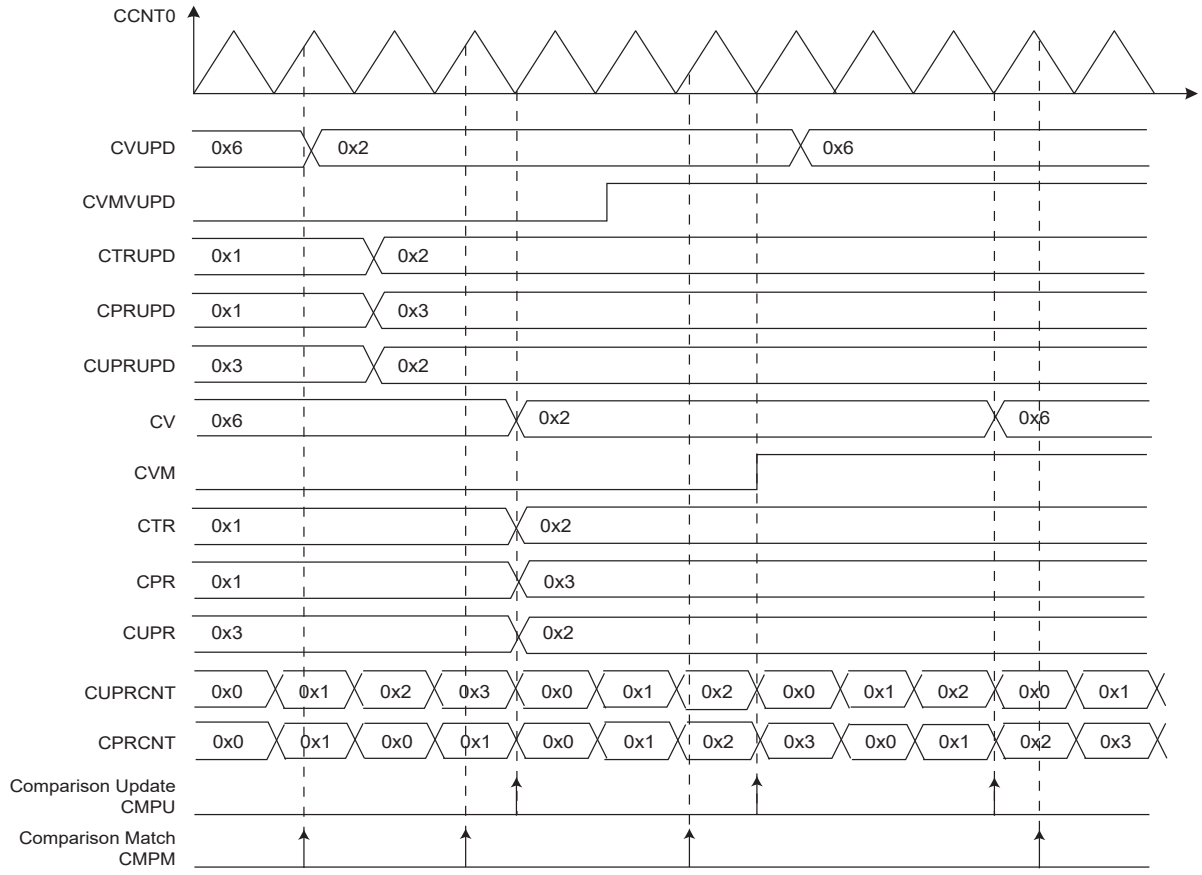
The comparison x configuration can be modified while the channel 0 is enabled by using the PWM Comparison x Mode Update Register (PWM_CMPMUPDx registers for the comparison x). In the same way, the comparison x value can be modified while the channel 0 is enabled by using the PWM Comparison x Value Update Register (PWM_CMPVUPDx registers for the comparison x).

The update of the comparison x configuration and the comparison x value is triggered periodically after the comparison x update period. It is defined by the field CUPR in PWM_CMPMx. The comparison unit has an update period counter independent from the period counter to trigger this update. When the value of the comparison update period counter CUPRCNT (in PWM_CMPMx) reaches the value defined by CUPR, the update is triggered. The comparison x update period CUPR itself can be updated while the channel 0 is enabled by using the PWM_CMPMUPDx register.

**⚠ CAUTION** The write of PWM_CMPVUPDx must be followed by a write of PWM_CMPMUPDx.

The comparison match and the comparison update can be source of an interrupt, but only if it is enabled and not masked. These interrupts can be enabled by the PWM Interrupt Enable Register 2 and disabled by the PWM Interrupt Disable Register 2. The comparison match interrupt and the comparison update interrupt are reset by reading the PWM Interrupt Status Register 2.

**Figure 55-24. Comparison Waveform**



### 55.6.4 PWM Event Lines

The PWM provides 2 independent event lines intended to trigger actions in other peripherals (e.g., for the Analog-to-Digital Converter (ADC)).

A pulse (one cycle of the peripheral clock) is generated on an event line, when at least one of the selected comparisons is matching. The comparisons can be selected or unselected independently by the CSEL bits in the PWM Event Line x Register (PWM_ELMRx for the Event Line x).

An example of event generation is provided in the figure Event Line Generation Waveform (Example).

**Figure 55-25. Event Line Block Diagram**

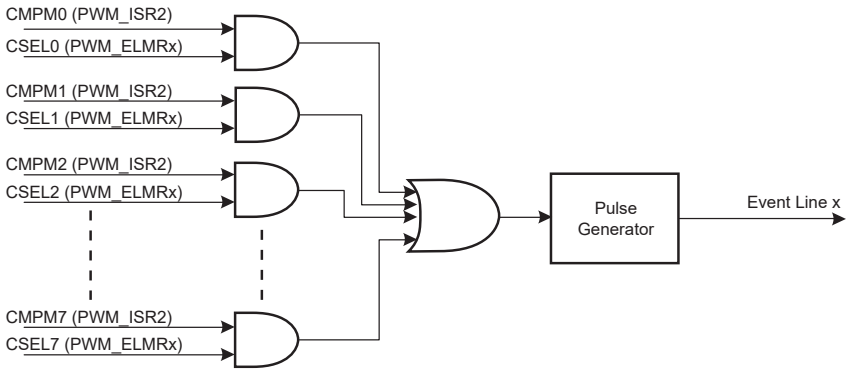
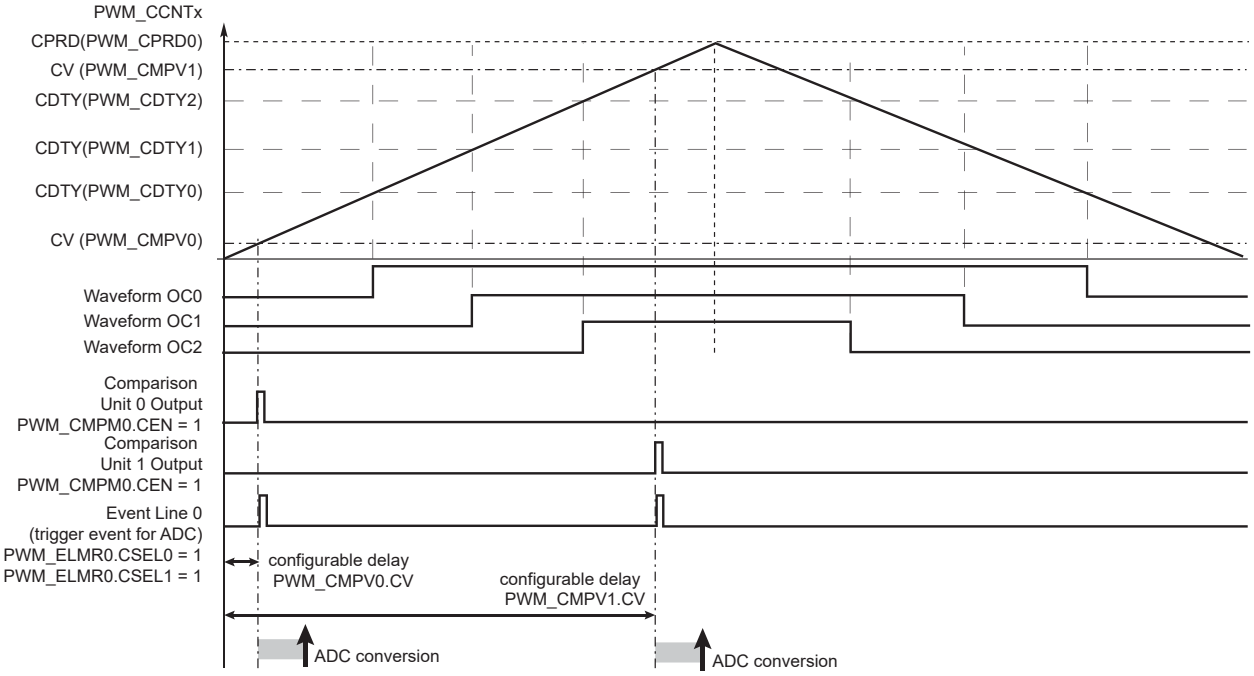



**Figure 55-26. Event Line Generation Waveform (Example)**




### 55.6.5 PWM External Trigger Mode

The PWM channels 1 and 2 can be configured to use an external trigger for generating specific PWM signals. The external trigger source can be selected through the bit TRGSRC of the PWM External Trigger Register (see the table below).

**Table 55-5. External Event Source Selection**

| Channel | Trigger Source Selection | Trigger Source |
|---|---|---|
| 1 | PWM_ETRG1.TRGSRC = 0 | From PWMEXTRG1 input |
| | PWM_ETRG1.TRGSRC = 1 | From Analog Comparator Controller |
| 2 | PWM_ETRG2.TRGSRC = 0 | From PWMEXTRG2 input |
| | PWM_ETRG2.TRGSRC = 1 | From Analog Comparator Controller |

Each external trigger source can be filtered by writing a one to the TRGFILT bit in the corresponding PWM External Trigger Register (PWM_ETRGx).

Each time an external trigger event is detected, the corresponding PWM channel counter value is stored in the MAXCNT field of the PWM_ETRGx register if it is greater than the previously stored value. Reading the PWM_ETRGx register will clear the MAXCNT value.

Three different modes are available for channels 1 and 2 depending on the value of the TRGMODE field of the PWM_ETRGx register:

- TRGMODE = 1: External PWM Reset Mode
- TRGMODE = 2: External PWM Start Mode
- TRGMODE = 3: Cycle-By-Cycle Duty Mode

See the following sections.

This feature is disabled when TRGMODE = 0.

This feature should only be enabled if the corresponding channel is left-aligned (CALG = 0 in PWM Channel Mode Register of channel 1 or 2) and not managed as a synchronous channel (SYNCx = 0 in PWM Sync Channels Mode Register where x = 1 or 2). Programming the channel to be center-aligned or synchronous while TRGMODE is not 0 could lead to unexpected behavior.

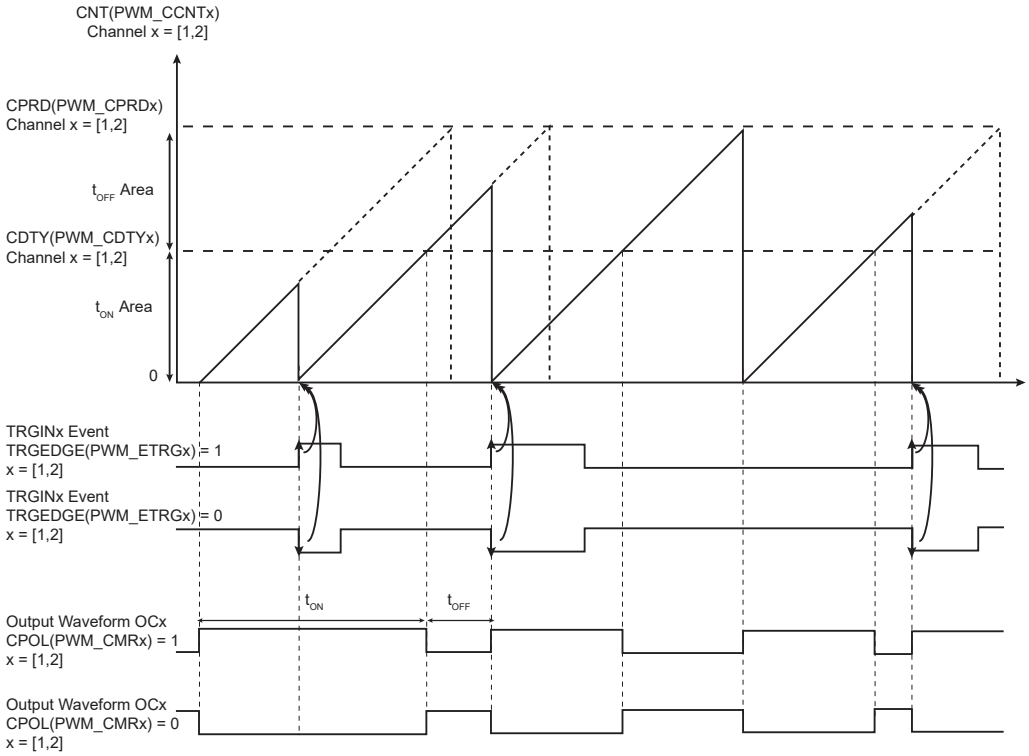
#### 55.6.5.1 External PWM Reset Mode

External PWM Reset mode is selected by programming TRGMODE = 1 in the PWM_ETRGx register.

In this mode, when an edge is detected on the PWMEXTRGx input, the internal PWM counter is cleared and a new PWM cycle is restarted. The edge polarity can be selected by programming the TRGEDGE bit in the PWM_ETRGx register. If no trigger event is detected when the internal channel counter has reached the CPRD value in the PWM Channel Period Register, the internal counter is cleared and a new PWM cycle starts.

Note that this mode does not guarantee a constant $t_{ON}$ or $t_{OFF}$ time.

**Figure 55-27. External PWM Reset Mode**



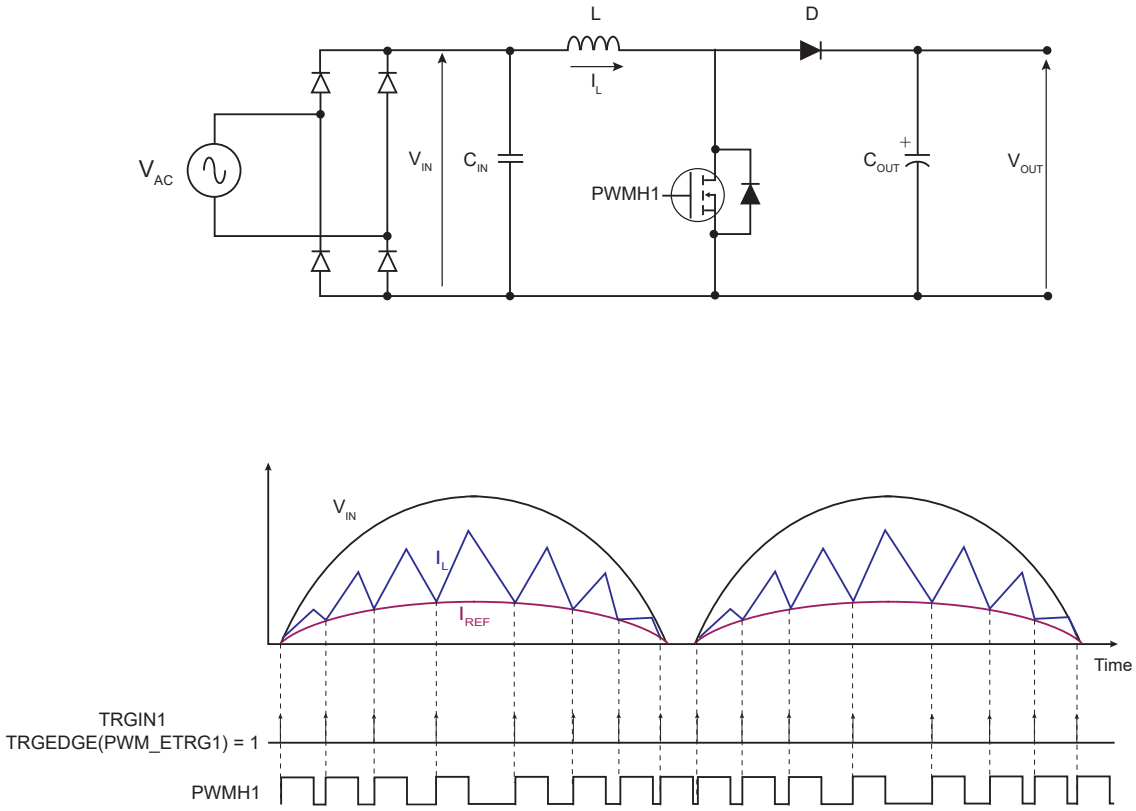

#### 55.6.5.1.1 Application Example

The external PWM Reset mode can be used in power factor correction applications.

In the example below, the external trigger input is the PWMEXTRG1 (therefore the PWM channel used for regulation is the channel 1). The PWM channel 1 period (CPRD in the PWM Channel Period Register of the channel 1) must be programmed so that the TRGIN1 event always triggers before the PWM channel 1 period elapses.

In the figure below, an external circuit (not shown) is required to sense the inductor current $I_L$. The internal PWM counter of the channel 1 is cleared when the inductor current falls below a specific threshold ($I_{REF}$). This starts a new PWM period and increases the inductor current.

**Figure 55-28. External PWM Reset Mode: Power Factor Correction Application**



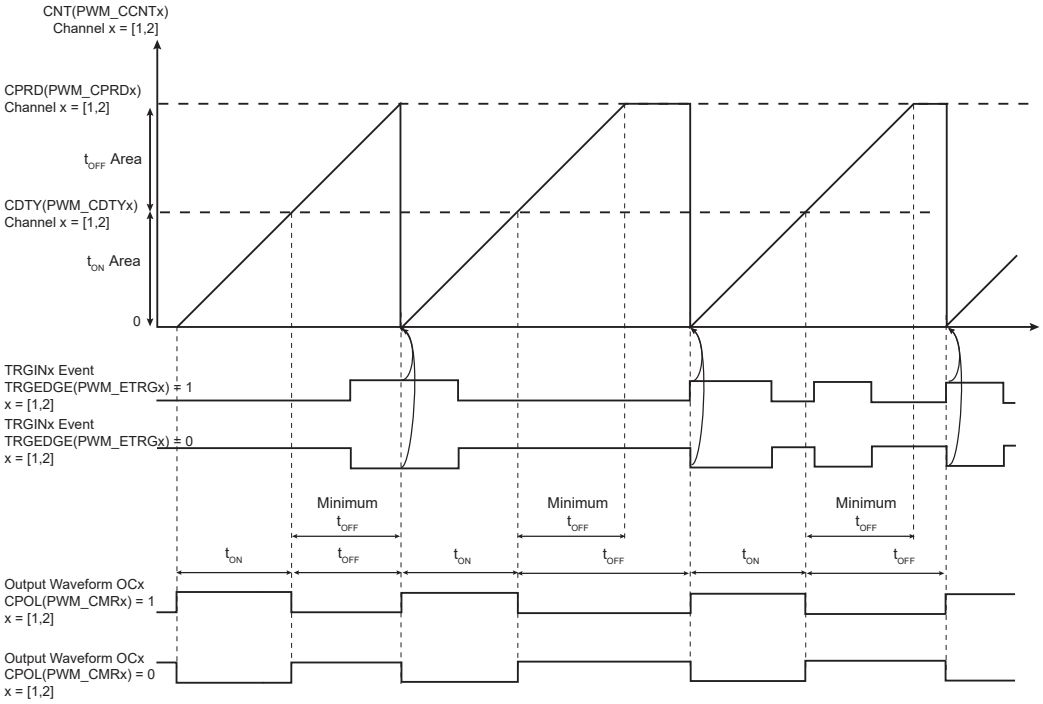

### 55.6.5.2 External PWM Start Mode

External PWM Start mode is selected by programming TRGMODE = 2 in the PWM_ETRGx register.

In this mode, the internal PWM counter can only be reset once it has reached the CPRD value in the PWM Channel Period Register and when the correct level is detected on the corresponding external trigger input. Both conditions have to be met to start a new PWM period. The active detection level is defined by the bit TRGEDGE of the PWM_ETRGx register.

Note that this mode guarantees a constant $t_{ON}$ time and a minimum $t_{OFF}$ time.

**Figure 55-29. External PWM Start Mode**



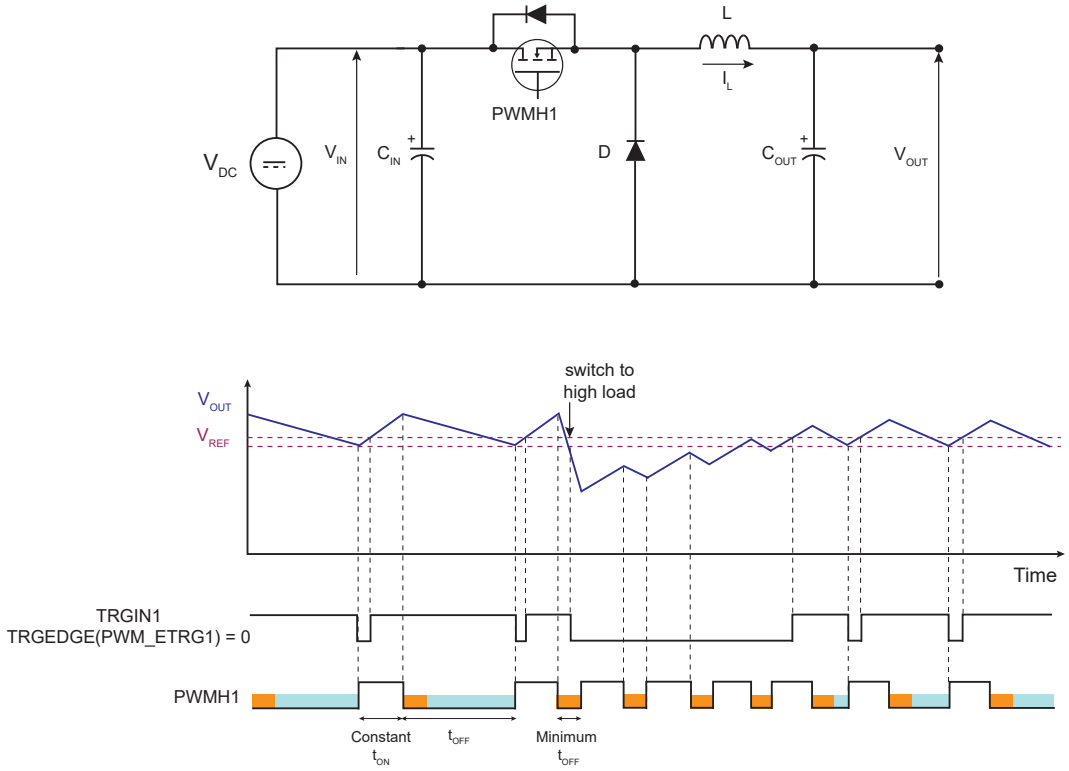

#### 55.6.5.2.1 Application Example

The external PWM Start mode generates a modulated frequency PWM signal with a constant active level duration ($t_{ON}$) and a minimum inactive level duration (minimum $t_{OFF}$).

The $t_{ON}$ time is defined by the CDTY value in the PWM Channel Duty Cycle Register. The minimum $t_{OFF}$ time is defined by CDTY - CPRD (PWM Channel Period Register). This mode can be useful in Buck DC/DC Converter applications.

When the output voltage $V_{OUT}$ is above a specific threshold (Vref), the PWM inactive level is maintained as long as $V_{OUT}$ remains above this threshold. If $V_{OUT}$ is below this specific threshold, this mode guarantees a minimum $t_{OFF}$ time required for MOSFET driving (see the figure below).

**Figure 55-30. External PWM Start Mode: Buck DC/DC Converter**




### 55.6.5.3 Cycle-By-Cycle Duty Mode

#### 55.6.5.3.1 Description

Cycle-by-cycle duty mode is selected by programming TRGMODE = 3 in PWM_ETRGx.

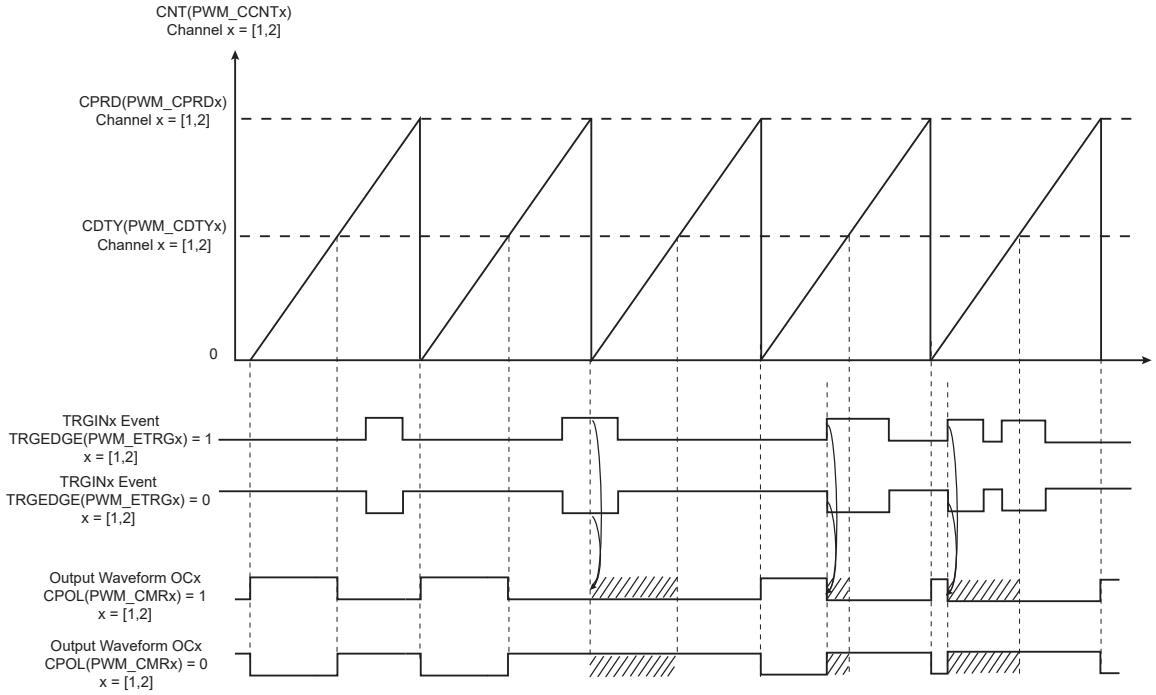
In this mode, the PWM frequency is constant and is defined by the CPRD value in the PWM Channel Period Register.

An external trigger event has no effect on the PWM output if it occurs while the internal PWM counter value is above the CDTY value of the PWM Channel Duty Cycle Register.

If the internal PWM counter value is below the value of CDTY of the PWM Channel Duty Cycle Register, an external trigger event makes the PWM output inactive.

The external trigger event can be detected on rising or falling edge according to the TRGEDGE bit in PWM_ETRGx.

**Figure 55-31. Cycle-By-Cycle Duty Mode**

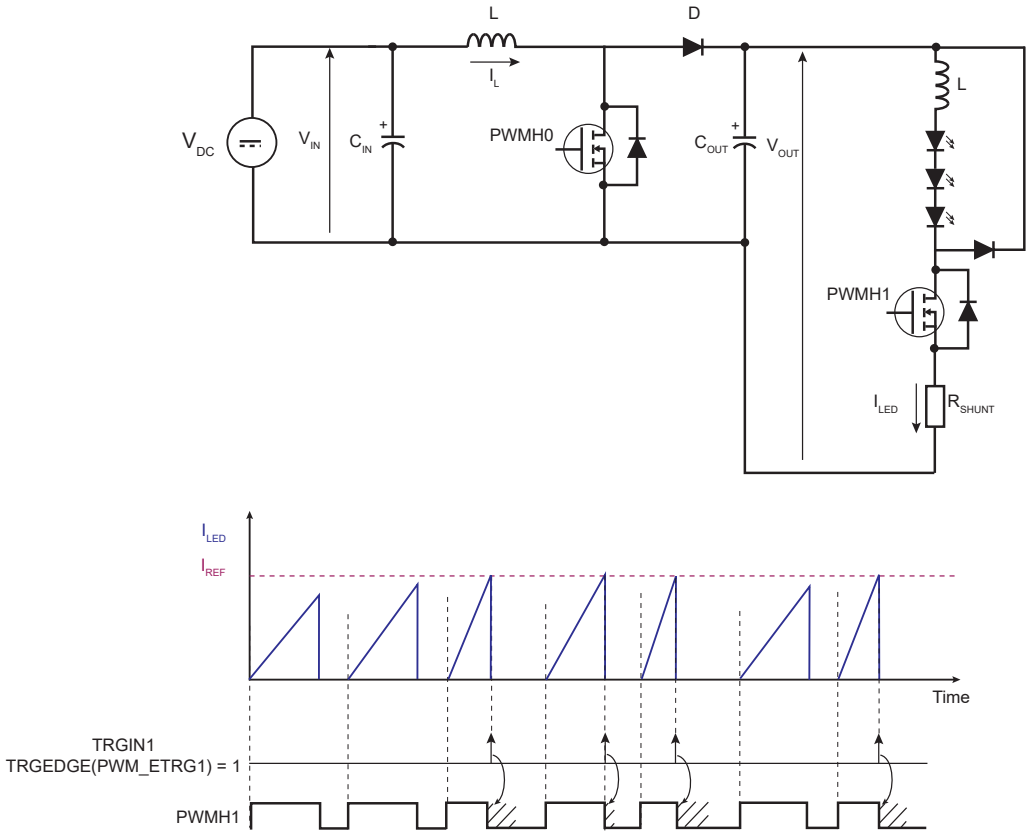



#### 55.6.5.3.2 Application Example

The figure below illustrates an application example of the Cycle-by-cycle Duty mode.

In an LED string control circuit, Cycle-by-cycle Duty mode can be used to automatically limit the current in the LED string.

**Figure 55-32. Cycle-By-Cycle Duty Mode: LED String Control**




### 55.6.5.4 Leading-Edge Blanking (LEB)

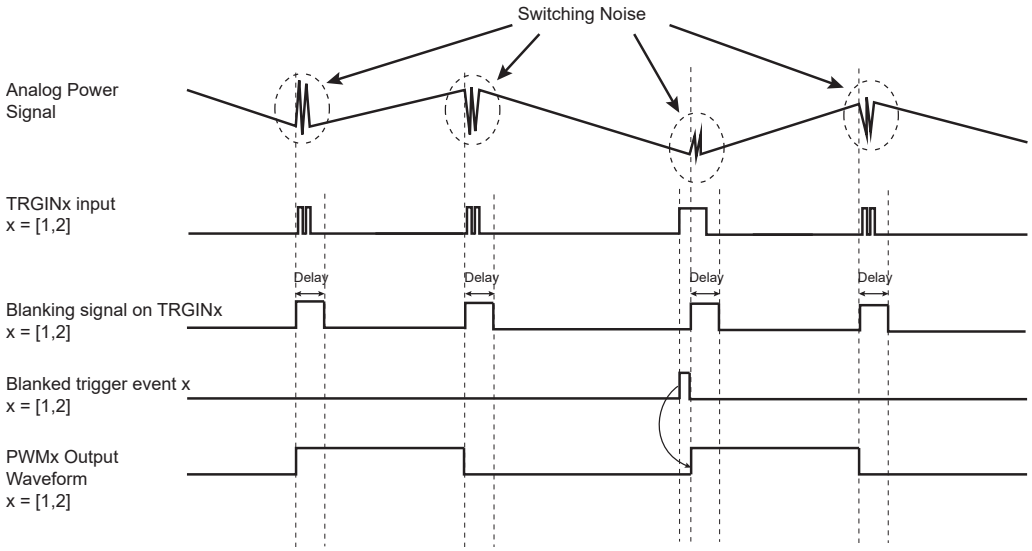
PWM channels 1 and 2 support leading-edge blanking. Leading-edge blanking masks the external trigger input when a transient occurs on the corresponding PWM output. It masks potential spurious external events due to power transistor switching.

The blanking delay on each external trigger input is configured by programming the LEBDELAYx in the PWM Leading-Edge Blanking Register.

The LEB can be enabled on both the rising and the falling edges for the PWMH and PWML outputs through the bits PWMLFEN, PWMLREN, PWMHFEN, PWMHREN.

Any event on the PWMEXTRGx input which occurs during the blanking time is ignored.

**Figure 55-33. Leading-Edge Blanking**




### 55.6.6 PWM Controller Operations

#### 55.6.6.1 Initialization

Before enabling the channels, they must be configured by the software application as described below:

- Unlock User Interface by writing the WPCMD field in PWM_WPCR.
- Configuration of the clock generator (DIVA, PREA, DIVB, PREB in the PWM_CLK register if required).
- Selection of the clock for each channel (CPRE field in PWM_CMRx)
- Configuration of the waveform alignment for each channel (CALG field in PWM_CMRx)
- Selection of the counter event selection (if CALG = 1) for each channel (CES field in PWM_CMRx)
- Configuration of the output waveform polarity for each channel (CPOL bit in PWM_CMRx)
- Configuration of the period for each channel (CPRD in the PWM_CPRDx register). Writing in PWM_CPRDx register is possible while the channel is disabled. After validation of the channel, the user must use PWM_CPRDUPDx register to update PWM_CPRDx as explained below.
- Configuration of the duty-cycle for each channel (CDTY in the PWM_CDTYx register). Writing in PWM_CDTYx register is possible while the channel is disabled. After validation of the channel, the user must use PWM_CDTYUPDx register to update PWM_CDTYx as explained below.
- Configuration of the dead-time generator for each channel (DTH and DTL in PWM_DTx) if enabled (DTE bit in PWM_CMRx). Writing in the PWM_DTx register is possible while the channel is disabled. After validation of the channel, the user must use PWM_DTUPDx register to update PWM_DTx
- Selection of the synchronous channels (SYNCx in the PWM_SCM register)
- Selection of the moment when the WRDY flag and the corresponding DMA Controller transfer request are set (PTRM and PTRCS in the PWM_SCM register)
- Configuration of the Update mode (UPDM in PWM_SCM register)
- Configuration of the update period (UPR in PWM_SCUP register) if needed
- Configuration of the comparisons (PWM_CMPVx and PWM_CMPMx)
- Configuration of the event lines (PWM_ELMRx)
- Configuration of the fault inputs polarity (FPOL in PWM_FMR)

- Configuration of the fault protection (FMOD and FFIL in PWM_FMR, PWM_FPV and PWM_FPE1)
- Enable of the interrupts (writing CHIDx and FCHIDx in PWM_IER1, and writing WRDY, UNRE, CMPMx and CMPUx in PWM_IER2)
- Enable of the PWM channels (writing CHIDx in the PWM_ENA register)

#### 55.6.6.2 Source Clock Selection Criteria

The large number of source clocks can make selection difficult. The relationship between the value in the PWM Channel Period Register (PWM_CPRDx) and the PWM Channel Duty Cycle Register (PWM_CDTYx) helps the user select the appropriate clock. The event number written in the Period Register gives the PWM accuracy. The Duty-Cycle quantum cannot be lower than 1/CPRDx value. The higher the value of PWM_CPRDx, the greater the PWM accuracy.

For example, if the user sets 15 (in decimal) in PWM_CPRDx, the user is able to set a value from between 1 up to 14 in PWM_CDTYx. The resulting duty-cycle quantum cannot be lower than 1/15 of the PWM period.

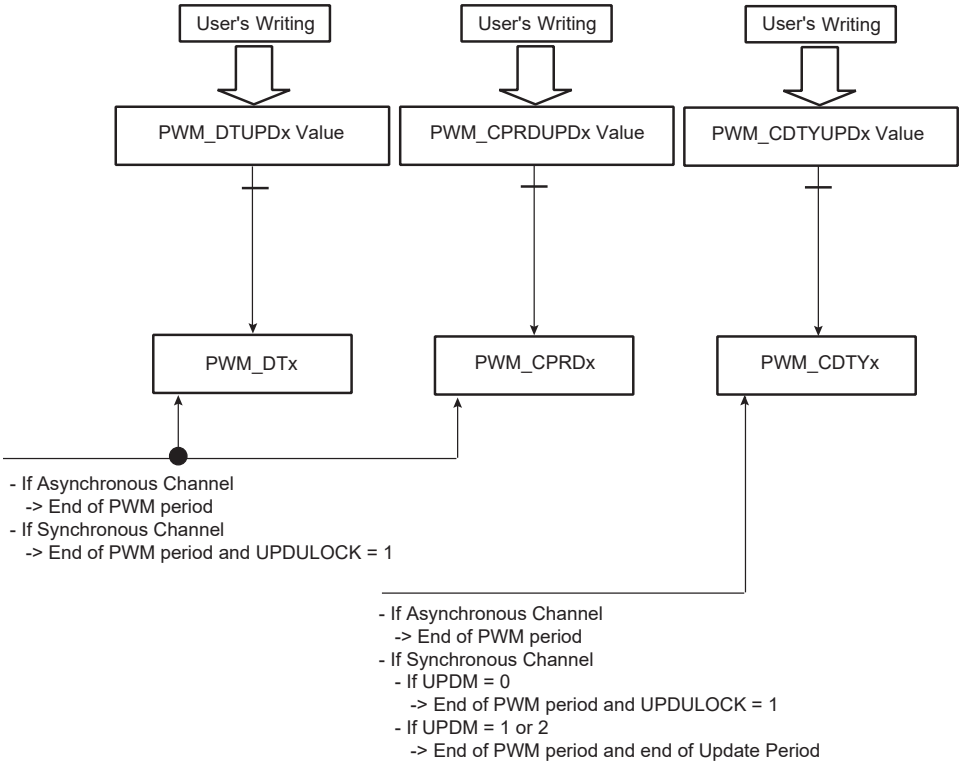
#### 55.6.6.3 Changing the Duty-Cycle, the Period and the Dead-Times

It is possible to modulate the output waveform duty-cycle, period and dead-times.

To prevent unexpected output waveform, the user must use the PWM Channel Duty Cycle Update Register (PWM_CDTYUPDx), the PWM Channel Period Update Register (PWM_CPRDUPDx) and the PWM Channel Dead Time Update Register (PWM_DTUPDx) to change waveform parameters while the channel is still enabled.

- If the channel is an asynchronous channel (SYNCx = 0 in PWM Sync Channels Mode Register (PWM_SCM)), these registers hold the new period, duty-cycle and dead-times values until the end of the current PWM period and update the values for the next period.
- If the channel is a synchronous channel and update method 0 is selected (SYNCx = 1 and UPDM = 0 in PWM_SCM register), these registers hold the new period, duty-cycle and dead-times values until the bit UPDULOCK is written at ‘1’ (in PWM Sync Channels Update Control Register (PWM_SCUC)) and the end of the current PWM period, then update the values for the next period.
- If the channel is a synchronous channel and update method 1 or 2 is selected (SYNCx = 1 and UPDM = 1 or 2 in PWM_SCM register):
  - registers PWM_CPRDUPDx and PWM_DTUPDx hold the new period and dead-times values until the bit UPDULOCK is written at ‘1’ (in PWM_SCUC) and the end of the current PWM period, then update the values for the next period.
  - register PWM_CDTYUPDx holds the new duty-cycle value until the end of the update period of synchronous channels (when UPRCNT is equal to UPR in PWM Sync Channels Update Period Register (PWM_SCUP)) and the end of the current PWM period, then updates the value for the next period.
    **Note:** If the update registers PWM_CDTYUPDx, PWM_CPRDUPDx and PWM_DTUPDx are written several times between two updates, only the last written value is taken into account.

**Figure 55-34. Synchronized Period, Duty-Cycle and Dead-Time Update**




### 55.6.6.4 Changing the Update Period of Synchronous Channels

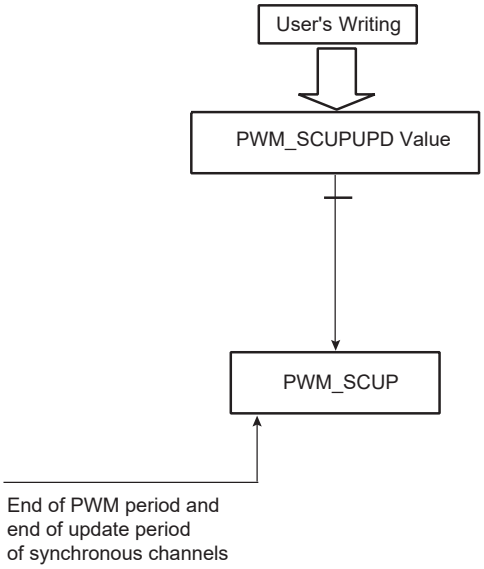
It is possible to change the update period of synchronous channels while they are enabled. See Method 2: Manual write of duty-cycle values and automatic trigger of the update and Method 3: Automatic write of duty-cycle values and automatic trigger of the update .

To prevent an unexpected update of the synchronous channels registers, the user must use the PWM Sync Channels Update Period Update Register (PWM_SCUPUPD) to change the update period of synchronous channels while they are still enabled. This register holds the new value until the end of the update period of synchronous channels (when UPRCNT is equal to UPR in PWM_SCUP) and the end of the current PWM period, then updates the value for the next period.

**Note:**

1. If the update register PWM_SCUPUPD is written several times between two updates, only the last written value is taken into account.
2. Changing the update period does make sense only if there is one or more synchronous channels and if the update method 1 or 2 is selected (UPDM = 1 or 2 in PWM Sync Channels Mode Register).

**Figure 55-35. Synchronized Update of Update Period Value of Synchronous Channels**




### 55.6.6.5 Changing the Comparison Value and the Comparison Configuration

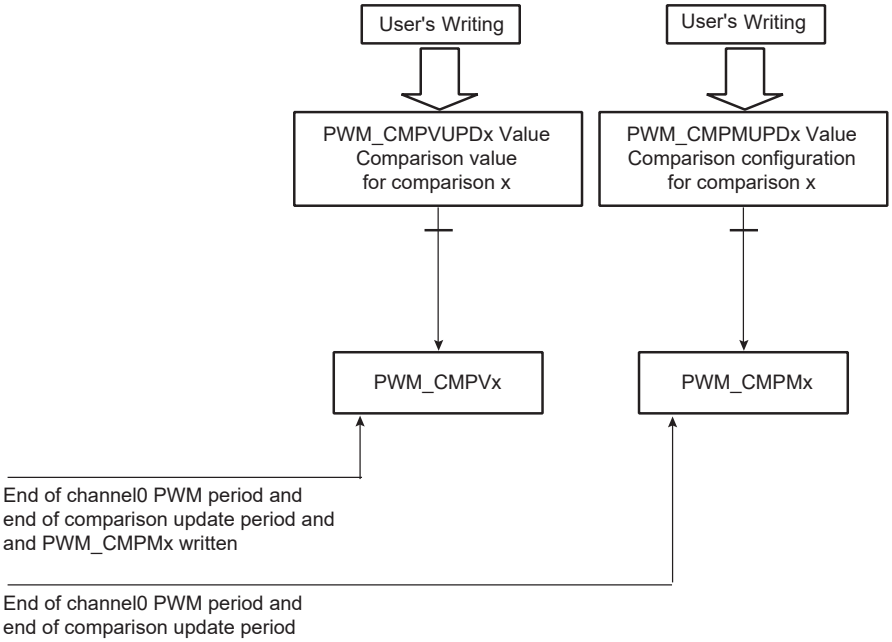
It is possible to change the comparison values and the comparison configurations while the channel 0 is enabled (see PWM Comparison Units).

To prevent unexpected comparison match, the user must use the PWM Comparison x Value Update Register (PWM_CMPVUPDx) and the PWM Comparison x Mode Update Register (PWM_CMPMUPDx) to change, respectively, the comparison values and the comparison configurations while the channel 0 is still enabled. These registers hold the new values until the end of the comparison update period (when CUPRCNT is equal to CUPR in PWM Comparison x Mode Register (PWM_CMPMx) and the end of the current PWM period, then update the values for the next period.

**⚠ CAUTION** The write of the register PWM_CMPVUPDx must be followed by a write of the register PWM_CMPMUPDx.

**Note:** If the update registers PWM_CMPVUPDx and PWM_CMPMUPDx are written several times between two updates, only the last written value are taken into account.

**Figure 55-36. Synchronized Update of Comparison Values and Configurations**




#### 55.6.6.6 Interrupt Sources

Depending on the interrupt mask in PWM_IMR1 and PWM_IMR2, an interrupt can be generated at the end of the corresponding channel period (CHIDx in the PWM Interrupt Status Register 1 (PWM_ISR1)), after a fault event (FCHIDx in PWM_ISR1), after a comparison match (CMPMx in PWM_ISR2), after a comparison update (CMPUx in PWM_ISR2) or according to the Transfer mode of the synchronous channels (WRDY and UNRE in PWM_ISR2).

If the interrupt is generated by the flags CHIDx or FCHIDx, the interrupt remains active until a read operation in PWM_ISR1 occurs.

If the interrupt is generated by the flags WRDY or UNRE or CMPMx or CMPUx, the interrupt remains active until a read operation in PWM_ISR2 occurs.

**A channel interrupt is enabled by setting the corresponding bit in PWM_IER1 and PWM_IER2. A channel interrupt is disabled by setting the corresponding bit in PWM_IDR1 and PWM_IDR2.**

### 55.6.7 Register Write Protection

To prevent any single software error that may corrupt PWM behavior, the registers listed below can be write-protected by writing the field WPCMD in the PWM Write Protection Control Register (PWM_WPCR). They are divided into six groups:

- Register group 0:
  - PWM Clock Register
- Register group 1:
  - PWM Disable Register
- Register group 2:
  - PWM Sync Channels Mode Register
  - PWM Channel Mode Register
  - PWM Stepper Motor Mode Register
  - PWM Fault Protection Value Register 2

  - PWM Leading-Edge Blanking Register
  - PWM Channel Mode Update Register
- Register group 3:
  - PWM Spread Spectrum Register
  - PWM Spread Spectrum Update Register
  - PWM Channel Period Register
  - PWM Channel Period Update Register
- Register group 4:
  - PWM Channel Dead Time Register
  - PWM Channel Dead Time Update Register
- Register group 5:
  - PWM Fault Mode Register
  - PWM Fault Protection Value Register 1

There are two types of write protection:

- SW write protection—can be enabled or disabled by software
- HW write protection—can be enabled by software but only disabled by a hardware reset of the PWM controller

Both types of write protection can be applied independently to a particular register group by means of the WPCMD and WPRGx fields in PWM_WPCR. If at least one type of write protection is active, the register group is write-protected. The value of field WPCMD defines the action to be performed:

- 0: Disables SW write protection of the register groups of which the bit WPRGx is at ‘1’
- 1: Enables SW write protection of the register groups of which the bit WPRGx is at ‘1’
- 2: Enables HW write protection of the register groups of which the bit WPRGx is at ‘1’

At any time, the user can determine whether SW or HW write protection is active in a particular register group by the fields WPSWS and WPHWS in the PWM Write Protection Status Register (PWM_WPSR).

If a write access to a write-protected register is detected, the WPVS flag in PWM_WPSR is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS and WPVSRC fields are automatically cleared after reading PWM_WPSR.

## 55.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | PWM_CLK | 7:0 | DIVA[7:0] | | | | | | | |
| | | 15:8 | | | | | PREA[3:0] | | | |
| | | 23:16 | DIVB[7:0] | | | | | | | |
| | | 31:24 | | | | | PREB[3:0] | | | |
| 0x04 | PWM_ENA | 7:0 | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | PWM_DIS | 7:0 | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | PWM_SR | 7:0 | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | PWM_IER1 | 7:0 | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | FCHID3 | FCHID2 | FCHID1 | FCHID0 |
| | | 31:24 | | | | | | | | |
| 0x14 | PWM_IDR1 | 7:0 | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | FCHID3 | FCHID2 | FCHID1 | FCHID0 |
| | | 31:24 | | | | | | | | |
| 0x18 | PWM_IMR1 | 7:0 | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | FCHID3 | FCHID2 | FCHID1 | FCHID0 |
| | | 31:24 | | | | | | | | |
| 0x1C | PWM_ISR1 | 7:0 | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | FCHID3 | FCHID2 | FCHID1 | FCHID0 |
| | | 31:24 | | | | | | | | |
| 0x20 | PWM_SCM | 7:0 | | | | | SYNC3 | SYNC2 | SYNC1 | SYNC0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | PTRCS[2:0] | | | PTRM | | | UPDM[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x24 | PWM_DMAR | 7:0 | DMADUTY[7:0] | | | | | | | |
| | | 15:8 | DMADUTY[15:8] | | | | | | | |
| | | 23:16 | DMADUTY[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x28 | PWM_SCUC | 7:0 | | | | | | | | UPDULOCK |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x2C | PWM_SCUP | 7:0 | UPRCNT[3:0] | | | | UPR[3:0] | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x30 | PWM_SCUPUPD | 7:0 | | | | | UPRUPD[3:0] | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x34 | PWM_IER2 | 7:0 | | | | | UNRE | | | WRDY |
| | | 15:8 | CMPM7 | CMPM6 | CMPM5 | CMPM4 | CMPM3 | CMPM2 | CMPM1 | CMPM0 |
| | | 23:16 | CMPU7 | CMPU6 | CMPU5 | CMPU4 | CMPU3 | CMPU2 | CMPU1 | CMPU0 |
| | | 31:24 | | | | | | | | |
| 0x38 | PWM_IDR2 | 7:0 | | | | | UNRE | | | WRDY |
| | | 15:8 | CMPM7 | CMPM6 | CMPM5 | CMPM4 | CMPM3 | CMPM2 | CMPM1 | CMPM0 |
| | | 23:16 | CMPU7 | CMPU6 | CMPU5 | CMPU4 | CMPU3 | CMPU2 | CMPU1 | CMPU0 |
| | | 31:24 | | | | | | | | |
| 0x3C | PWM_IMR2 | 7:0 | | | | | UNRE | | | WRDY |
| | | 15:8 | CMPM7 | CMPM6 | CMPM5 | CMPM4 | CMPM3 | CMPM2 | CMPM1 | CMPM0 |
| | | 23:16 | CMPU7 | CMPU6 | CMPU5 | CMPU4 | CMPU3 | CMPU2 | CMPU1 | CMPU0 |
| | | 31:24 | | | | | | | | |
| 0x40 | PWM_ISR2 | 7:0 | | | | | UNRE | | | WRDY |
| | | 15:8 | CMPM7 | CMPM6 | CMPM5 | CMPM4 | CMPM3 | CMPM2 | CMPM1 | CMPM0 |
| | | 23:16 | CMPU7 | CMPU6 | CMPU5 | CMPU4 | CMPU3 | CMPU2 | CMPU1 | CMPU0 |
| | | 31:24 | | | | | | | | |
| 0x44 | PWM_OOV | 7:0 | | | | | OOVH3 | OOVH2 | OOVH1 | OOVH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | OOVL3 | OOVL2 | OOVL1 | OOVL0 |
| | | 31:24 | | | | | | | | |
| 0x48 | PWM_OS | 7:0 | | | | | OSH3 | OSH2 | OSH1 | OSH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | OSL3 | OSL2 | OSL1 | OSL0 |
| | | 31:24 | | | | | | | | |
| 0x4C | PWM_OSS | 7:0 | | | | | OSSH3 | OSSH2 | OSSH1 | OSSH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | OSSL3 | OSSL2 | OSSL1 | OSSL0 |
| | | 31:24 | | | | | | | | |
| 0x50 | PWM_OSC | 7:0 | | | | | OSCH3 | OSCH2 | OSCH1 | OSCH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | OSCL3 | OSCL2 | OSCL1 | OSCL0 |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x54 | PWM_OSSUPD | 7:0 | | | | | OSSUPH3 | OSSUPH2 | OSSUPH1 | OSSUPH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | OSSUPL3 | OSSUPL2 | OSSUPL1 | OSSUPL0 |
| | | 31:24 | | | | | | | | |
| 0x58 | PWM_OSCUPD | 7:0 | | | | | OSCUPH3 | OSCUPH2 | OSCUPH1 | OSCUPH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | OSCUPL3 | OSCUPL2 | OSCUPL1 | OSCUPL0 |
| | | 31:24 | | | | | | | | |
| 0x5C | PWM_FMR | 7:0 | FPOL[7:0] | | | | | | | |
| | | 15:8 | FMOD[7:0] | | | | | | | |
| | | 23:16 | FFIL[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x60 | PWM_FSR | 7:0 | FIV[7:0] | | | | | | | |
| | | 15:8 | FS[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x64 | PWM_FCR | 7:0 | FCLR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x68 | PWM_FPV1 | 7:0 | | | | | FPVH3 | FPVH2 | FPVH1 | FPVH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | FPVL3 | FPVL2 | FPVL1 | FPVL0 |
| | | 31:24 | | | | | | | | |
| 0x6C | PWM_FPE | 7:0 | FPE0[7:0] | | | | | | | |
| | | 15:8 | FPE1[7:0] | | | | | | | |
| | | 23:16 | FPE2[7:0] | | | | | | | |
| | | 31:24 | FPE3[7:0] | | | | | | | |
| 0x70 ... 0x7B | Reserved | | | | | | | | | |
| 0x7C | PWM_ELMR0 | 7:0 | CSEL7 | CSEL6 | CSEL5 | CSEL4 | CSEL3 | CSEL2 | CSEL1 | CSEL0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x80 | PWM_ELMR1 | 7:0 | CSEL7 | CSEL6 | CSEL5 | CSEL4 | CSEL3 | CSEL2 | CSEL1 | CSEL0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x84 ... 0x9F | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xA0 | PWM_SSPR | 7:0 | SPRD[7:0] | | | | | | | |
| | | 15:8 | SPRD[15:8] | | | | | | | |
| | | 23:16 | SPRD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | SPRDM |
| 0xA4 | PWM_SSPUP | 7:0 | SPRDUP[7:0] | | | | | | | |
| | | 15:8 | SPRDUP[15:8] | | | | | | | |
| | | 23:16 | SPRDUP[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xA8 ... 0xAF | Reserved | | | | | | | | | |
| 0xB0 | PWM_SMMR | 7:0 | | | | | | | GCEN1 | GCEN0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | DOWN1 | DOWN0 |
| | | 31:24 | | | | | | | | |
| 0xB4 ... 0xBF | Reserved | | | | | | | | | |
| 0xC0 | PWM_FPV2 | 7:0 | | | | | FPZH3 | FPZH2 | FPZH1 | FPZH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | FPZL3 | FPZL2 | FPZL1 | FPZL0 |
| | | 31:24 | | | | | | | | |
| 0xC4 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | PWM_WPCR | 7:0 | WPRG5 | WPRG4 | WPRG3 | WPRG2 | WPRG1 | WPRG0 | WPCMD[1:0] | |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | PWM_WPSR | 7:0 | WPVS | | WPSWS5 | WPSWS4 | WPSWS3 | WPSWS2 | WPSWS1 | WPSWS0 |
| | | 15:8 | | | WPHWS5 | WPHWS4 | WPHWS3 | WPHWS2 | WPHWS1 | WPHWS0 |
| | | 23:16 | WPVSRC[7:0] | | | | | | | |
| | | 31:24 | WPVSRC[15:8] | | | | | | | |
| 0xEC ... 0x012F | Reserved | | | | | | | | | |
| 0x0130 | PWM_CMPV0 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVM |
| 0x0134 | PWM_CMPVUPD0 | 7:0 | CVUPD[7:0] | | | | | | | |
| | | 15:8 | CVUPD[15:8] | | | | | | | |
| | | 23:16 | CVUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVMUPD |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0138 | PWM_CMPM0 | 7:0 | CTR[3:0] | | | | | | | CEN |
| | | 15:8 | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| | | 23:16 | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x013C | PWM_CMPMUPD0 | 7:0 | CTRUPD[3:0] | | | | | | | CENUPD |
| | | 15:8 | | | | | CPRUPD[3:0] | | | |
| | | 23:16 | | | | | CUPRUPD[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x0140 | PWM_CMPV1 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVM |
| 0x0144 | PWM_CMPVUPD1 | 7:0 | CVUPD[7:0] | | | | | | | |
| | | 15:8 | CVUPD[15:8] | | | | | | | |
| | | 23:16 | CVUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVMUPD |
| 0x0148 | PWM_CMPM1 | 7:0 | CTR[3:0] | | | | | | | CEN |
| | | 15:8 | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| | | 23:16 | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x014C | PWM_CMPMUPD1 | 7:0 | CTRUPD[3:0] | | | | | | | CENUPD |
| | | 15:8 | | | | | CPRUPD[3:0] | | | |
| | | 23:16 | | | | | CUPRUPD[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x0150 | PWM_CMPV2 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVM |
| 0x0154 | PWM_CMPVUPD2 | 7:0 | CVUPD[7:0] | | | | | | | |
| | | 15:8 | CVUPD[15:8] | | | | | | | |
| | | 23:16 | CVUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVMUPD |
| 0x0158 | PWM_CMPM2 | 7:0 | CTR[3:0] | | | | | | | CEN |
| | | 15:8 | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| | | 23:16 | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x015C | PWM_CMPMUPD2 | 7:0 | CTRUPD[3:0] | | | | | | | CENUPD |
| | | 15:8 | | | | | CPRUPD[3:0] | | | |
| | | 23:16 | | | | | CUPRUPD[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x0160 | PWM_CMPV3 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVM |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0164 | PWM_CMPVUPD3 | 7:0 | CVUPD[7:0] | | | | | | | |
| | | 15:8 | CVUPD[15:8] | | | | | | | |
| | | 23:16 | CVUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVMUPD |
| 0x0168 | PWM_CMPM3 | 7:0 | CTR[3:0] | | | | | | | CEN |
| | | 15:8 | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| | | 23:16 | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x016C | PWM_CMPMUPD3 | 7:0 | CTRUPD[3:0] | | | | | | | CENUPD |
| | | 15:8 | | | | | CPRUPD[3:0] | | | |
| | | 23:16 | | | | | CUPRUPD[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x0170 | PWM_CMPV4 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVM |
| 0x0174 | PWM_CMPVUPD4 | 7:0 | CVUPD[7:0] | | | | | | | |
| | | 15:8 | CVUPD[15:8] | | | | | | | |
| | | 23:16 | CVUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVMUPD |
| 0x0178 | PWM_CMPM4 | 7:0 | CTR[3:0] | | | | | | | CEN |
| | | 15:8 | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| | | 23:16 | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x017C | PWM_CMPMUPD4 | 7:0 | CTRUPD[3:0] | | | | | | | CENUPD |
| | | 15:8 | | | | | CPRUPD[3:0] | | | |
| | | 23:16 | | | | | CUPRUPD[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x0180 | PWM_CMPV5 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVM |
| 0x0184 | PWM_CMPVUPD5 | 7:0 | CVUPD[7:0] | | | | | | | |
| | | 15:8 | CVUPD[15:8] | | | | | | | |
| | | 23:16 | CVUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVMUPD |
| 0x0188 | PWM_CMPM5 | 7:0 | CTR[3:0] | | | | | | | CEN |
| | | 15:8 | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| | | 23:16 | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x018C | PWM_CMPMUPD5 | 7:0 | CTRUPD[3:0] | | | | | | | CENUPD |
| | | 15:8 | | | | | CPRUPD[3:0] | | | |
| | | 23:16 | | | | | CUPRUPD[3:0] | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0190 | PWM_CMPV6 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVM |
| 0x0194 | PWM_CMPVUPD6 | 7:0 | CVUPD[7:0] | | | | | | | |
| | | 15:8 | CVUPD[15:8] | | | | | | | |
| | | 23:16 | CVUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVMUPD |
| 0x0198 | PWM_CMPM6 | 7:0 | CTR[3:0] | | | | | | | CEN |
| | | 15:8 | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| | | 23:16 | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x019C | PWM_CMPMUPD6 | 7:0 | CTRUPD[3:0] | | | | | | | CENUPD |
| | | 15:8 | | | | | CPRUPD[3:0] | | | |
| | | 23:16 | | | | | CUPRUPD[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x01A0 | PWM_CMPV7 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVM |
| 0x01A4 | PWM_CMPVUPD7 | 7:0 | CVUPD[7:0] | | | | | | | |
| | | 15:8 | CVUPD[15:8] | | | | | | | |
| | | 23:16 | CVUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVMUPD |
| 0x01A8 | PWM_CMPM7 | 7:0 | CTR[3:0] | | | | | | | CEN |
| | | 15:8 | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| | | 23:16 | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x01AC | PWM_CMPMUPD7 | 7:0 | CTRUPD[3:0] | | | | | | | CENUPD |
| | | 15:8 | | | | | CPRUPD[3:0] | | | |
| | | 23:16 | | | | | CUPRUPD[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x01B0 ... 0x01FF | Reserved | | | | | | | | | |
| 0x0200 | PWM_CMR0 | 7:0 | | | | | CPRE[3:0] | | | |
| | | 15:8 | | | TCTS | DPOLI | UPDS | CES | CPOL | CALG |
| | | 23:16 | | | | | PPM | DTLI | DTHI | DTE |
| | | 31:24 | | | | | | | | |
| 0x0204 | PWM_CDTY0 | 7:0 | CDTY[7:0] | | | | | | | |
| | | 15:8 | CDTY[15:8] | | | | | | | |
| | | 23:16 | CDTY[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0208 | PWM_CDTYUPD0 | 7:0 | CDTYUPD[7:0] | | | | | | | |
| | | 15:8 | CDTYUPD[15:8] | | | | | | | |
| | | 23:16 | CDTYUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x020C | PWM_CPRD0 | 7:0 | CPRD[7:0] | | | | | | | |
| | | 15:8 | CPRD[15:8] | | | | | | | |
| | | 23:16 | CPRD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0210 | PWM_CPRDUPD0 | 7:0 | CPRDUPD[7:0] | | | | | | | |
| | | 15:8 | CPRDUPD[15:8] | | | | | | | |
| | | 23:16 | CPRDUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0214 | PWM_CCNT0 | 7:0 | CNT[7:0] | | | | | | | |
| | | 15:8 | CNT[15:8] | | | | | | | |
| | | 23:16 | CNT[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0218 | PWM_DT0 | 7:0 | DTH[7:0] | | | | | | | |
| | | 15:8 | DTH[15:8] | | | | | | | |
| | | 23:16 | DTL[7:0] | | | | | | | |
| | | 31:24 | DTL[15:8] | | | | | | | |
| 0x021C | PWM_DTUPD0 | 7:0 | DTHUPD[7:0] | | | | | | | |
| | | 15:8 | DTHUPD[15:8] | | | | | | | |
| | | 23:16 | DTLUPD[7:0] | | | | | | | |
| | | 31:24 | DTLUPD[15:8] | | | | | | | |
| 0x0220 | PWM_CMR1 | 7:0 | | | | | CPRE[3:0] | | | |
| | | 15:8 | | | TCTS | DPOLI | UPDS | CES | CPOL | CALG |
| | | 23:16 | | | | | PPM | DTLI | DTHI | DTE |
| | | 31:24 | | | | | | | | |
| 0x0224 | PWM_CDTY1 | 7:0 | CDTY[7:0] | | | | | | | |
| | | 15:8 | CDTY[15:8] | | | | | | | |
| | | 23:16 | CDTY[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0228 | PWM_CDTYUPD1 | 7:0 | CDTYUPD[7:0] | | | | | | | |
| | | 15:8 | CDTYUPD[15:8] | | | | | | | |
| | | 23:16 | CDTYUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x022C | PWM_CPRD1 | 7:0 | CPRD[7:0] | | | | | | | |
| | | 15:8 | CPRD[15:8] | | | | | | | |
| | | 23:16 | CPRD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0230 | PWM_CPRDUPD1 | 7:0 | CPRDUPD[7:0] | | | | | | | |
| | | 15:8 | CPRDUPD[15:8] | | | | | | | |
| | | 23:16 | CPRDUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0234 | PWM_CCNT1 | 7:0 | CNT[7:0] | | | | | | | |
| | | 15:8 | CNT[15:8] | | | | | | | |
| | | 23:16 | CNT[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0238 | PWM_DT1 | 7:0 | DTH[7:0] | | | | | | | |
| | | 15:8 | DTH[15:8] | | | | | | | |
| | | 23:16 | DTL[7:0] | | | | | | | |
| | | 31:24 | DTL[15:8] | | | | | | | |
| 0x023C | PWM_DTUPD1 | 7:0 | DTHUPD[7:0] | | | | | | | |
| | | 15:8 | DTHUPD[15:8] | | | | | | | |
| | | 23:16 | DTLUPD[7:0] | | | | | | | |
| | | 31:24 | DTLUPD[15:8] | | | | | | | |
| 0x0240 | PWM_CMR2 | 7:0 | | | | | CPRE[3:0] | | | |
| | | 15:8 | | | TCTS | DPOLI | UPDS | CES | CPOL | CALG |
| | | 23:16 | | | | | PPM | DTLI | DTHI | DTE |
| | | 31:24 | | | | | | | | |
| 0x0244 | PWM_CDTY2 | 7:0 | CDTY[7:0] | | | | | | | |
| | | 15:8 | CDTY[15:8] | | | | | | | |
| | | 23:16 | CDTY[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0248 | PWM_CDTYUPD2 | 7:0 | CDTYUPD[7:0] | | | | | | | |
| | | 15:8 | CDTYUPD[15:8] | | | | | | | |
| | | 23:16 | CDTYUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x024C | PWM_CPRD2 | 7:0 | CPRD[7:0] | | | | | | | |
| | | 15:8 | CPRD[15:8] | | | | | | | |
| | | 23:16 | CPRD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0250 | PWM_CPRDUPD2 | 7:0 | CPRDUPD[7:0] | | | | | | | |
| | | 15:8 | CPRDUPD[15:8] | | | | | | | |
| | | 23:16 | CPRDUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0254 | PWM_CCNT2 | 7:0 | CNT[7:0] | | | | | | | |
| | | 15:8 | CNT[15:8] | | | | | | | |
| | | 23:16 | CNT[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0258 | PWM_DT2 | 7:0 | DTH[7:0] | | | | | | | |
| | | 15:8 | DTH[15:8] | | | | | | | |
| | | 23:16 | DTL[7:0] | | | | | | | |
| | | 31:24 | DTL[15:8] | | | | | | | |
| 0x025C | PWM_DTUPD2 | 7:0 | DTHUPD[7:0] | | | | | | | |
| | | 15:8 | DTHUPD[15:8] | | | | | | | |
| | | 23:16 | DTLUPD[7:0] | | | | | | | |
| | | 31:24 | DTLUPD[15:8] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0260 | PWM_CMR3 | 7:0 | | | | | CPRE[3:0] | | | |
| | | 15:8 | | | TCTS | DPOLI | UPDS | CES | CPOL | CALG |
| | | 23:16 | | | | | PPM | DTLI | DTHI | DTE |
| | | 31:24 | | | | | | | | |
| 0x0264 | PWM_CDTY3 | 7:0 | CDTY[7:0] | | | | | | | |
| | | 15:8 | CDTY[15:8] | | | | | | | |
| | | 23:16 | CDTY[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0268 | PWM_CDTYUPD3 | 7:0 | CDTYUPD[7:0] | | | | | | | |
| | | 15:8 | CDTYUPD[15:8] | | | | | | | |
| | | 23:16 | CDTYUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x026C | PWM_CPRD3 | 7:0 | CPRD[7:0] | | | | | | | |
| | | 15:8 | CPRD[15:8] | | | | | | | |
| | | 23:16 | CPRD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0270 | PWM_CPRDUPD3 | 7:0 | CPRDUPD[7:0] | | | | | | | |
| | | 15:8 | CPRDUPD[15:8] | | | | | | | |
| | | 23:16 | CPRDUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0274 | PWM_CCNT3 | 7:0 | CNT[7:0] | | | | | | | |
| | | 15:8 | CNT[15:8] | | | | | | | |
| | | 23:16 | CNT[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0278 | PWM_DT3 | 7:0 | DTH[7:0] | | | | | | | |
| | | 15:8 | DTH[15:8] | | | | | | | |
| | | 23:16 | DTL[7:0] | | | | | | | |
| | | 31:24 | DTL[15:8] | | | | | | | |
| 0x027C | PWM_DTUPD3 | 7:0 | DTHUPD[7:0] | | | | | | | |
| | | 15:8 | DTHUPD[15:8] | | | | | | | |
| | | 23:16 | DTLUPD[7:0] | | | | | | | |
| | | 31:24 | DTLUPD[15:8] | | | | | | | |
| 0x0280 ... 0x03FF | Reserved | | | | | | | | | |
| 0x0400 | PWM_CMUPD0 | 7:0 | | | | | | | | |
| | | 15:8 | | | CPOLINVUP | | | | CPOLUP | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0404 ... 0x041F | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0420 | PWM_CMUPD1 | 7:0 | | | | | | | | |
| | | 15:8 | | | CPOLINVUP | | | | CPOLUP | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0424 ... 0x042B | Reserved | | | | | | | | | |
| 0x042C | PWM_ETRG1 | 7:0 | MAXCNT[7:0] | | | | | | | |
| | | 15:8 | MAXCNT[15:8] | | | | | | | |
| | | 23:16 | MAXCNT[23:16] | | | | | | | |
| | | 31:24 | RFEN | TRGSRC | TRGFILT | TRGEDGE | | | TRGMODE[1:0] | |
| 0x0430 | PWM_LEBR1 | 7:0 | | LEBDELAY[6:0] | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | PWMHREN | PWMHFEN | PWMLREN | PWMLFEN |
| | | 31:24 | | | | | | | | |
| 0x0434 ... 0x043F | Reserved | | | | | | | | | |
| 0x0440 | PWM_CMUPD2 | 7:0 | | | | | | | | |
| | | 15:8 | | | CPOLINVUP | | | | CPOLUP | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0444 ... 0x044B | Reserved | | | | | | | | | |
| 0x044C | PWM_ETRG2 | 7:0 | MAXCNT[7:0] | | | | | | | |
| | | 15:8 | MAXCNT[15:8] | | | | | | | |
| | | 23:16 | MAXCNT[23:16] | | | | | | | |
| | | 31:24 | RFEN | TRGSRC | TRGFILT | TRGEDGE | | | TRGMODE[1:0] | |
| 0x0450 | PWM_LEBR2 | 7:0 | | LEBDELAY[6:0] | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | PWMHREN | PWMHFEN | PWMLREN | PWMLFEN |
| | | 31:24 | | | | | | | | |
| 0x0454 ... 0x045F | Reserved | | | | | | | | | |
| 0x0460 | PWM_CMUPD3 | 7:0 | | | | | | | | |
| | | 15:8 | | | CPOLINVUP | | | | CPOLUP | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 55.7.1 PWM Clock Register

**Name:** PWM_CLK
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS0 and WPHWS0 are cleared in the PWM Write Protection Status Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PREB[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DIVB[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PREA[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DIVA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27:24 – PREB[3:0]** CLKB Source Clock Selection

| Value | Name | Description |
|---|---|---|
| 0 | CLK | Peripheral clock |
| 1 | CLK_DIV2 | Peripheral clock/2 |
| 2 | CLK_DIV4 | Peripheral clock/4 |
| 3 | CLK_DIV8 | Peripheral clock/8 |
| 4 | CLK_DIV16 | Peripheral clock/16 |
| 5 | CLK_DIV32 | Peripheral clock/32 |
| 6 | CLK_DIV64 | Peripheral clock/64 |
| 7 | CLK_DIV128 | Peripheral clock/128 |
| 8 | CLK_DIV256 | Peripheral clock/256 |
| 9 | CLK_DIV512 | Peripheral clock/512 |
| 10 | CLK_DIV1024 | Peripheral clock/1024 |
| Other | – | Reserved |

**Bits 23:16 – DIVB[7:0]** CLKB Divide Factor

| Value | Name | Description |
|---|---|---|
| 0 | CLKB_POFF | CLKB clock is turned off |
| 1 | PREB | CLKB clock is clock selected by PREB |
| 2-255 | PREB_DIV | CLKB clock is clock selected by PREB divided by DIVB factor |

**Bits 11:8 – PREA[3:0]** CLKA Source Clock Selection

| Value | Name | Description |
|---|---|---|
| 0 | CLK | Peripheral clock |
| 1 | CLK_DIV2 | Peripheral clock/2 |
| 2 | CLK_DIV4 | Peripheral clock/4 |
| 3 | CLK_DIV8 | Peripheral clock/8 |
| 4 | CLK_DIV16 | Peripheral clock/16 |
| 5 | CLK_DIV32 | Peripheral clock/32 |
| 6 | CLK_DIV64 | Peripheral clock/64 |
| 7 | CLK_DIV128 | Peripheral clock/128 |
| 8 | CLK_DIV256 | Peripheral clock/256 |
| 9 | CLK_DIV512 | Peripheral clock/512 |
| 10 | CLK_DIV1024 | Peripheral clock/1024 |
| Other | – | Reserved |

**Bits 7:0 – DIVA[7:0]** CLKA Divide Factor

| Value | Name | Description |
|---|---|---|
| 0 | CLKA_POFF | CLKA clock is turned off |
| 1 | PREA | CLKA clock is clock selected by PREA |
| 2-255 | PREA_DIV | CLKA clock is clock selected by PREA divided by DIVA factor |

### 55.7.2 PWM Enable Register

**Name:** PWM_ENA
**Offset:** 0x04
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 0, 1, 2, 3 – CHIDx** Channel ID

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable PWM output for channel x. |

### 55.7.3 PWM Disable Register

**Name:** PWM_DIS
**Offset:** 0x08
**Reset:** –
**Property:** Write-only

This register can only be written if bits WPSWS1 and WPHWS1 are cleared in the PWM Write Protection Status Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 0, 1, 2, 3 – CHIDx** Channel ID

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable PWM output for channel x. |

### 55.7.4 PWM Status Register

**Name:** PWM_SR
**Offset:** 0x0C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3 – CHIDx** Channel ID

| Value | Description |
|---|---|
| 0 | PWM output for channel x is disabled. |
| 1 | PWM output for channel x is enabled. |

### 55.7.5 PWM Interrupt Enable Register 1

**Name:** PWM_IER1
**Offset:** 0x10
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FCHID3 | FCHID2 | FCHID1 | FCHID0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 16, 17, 18, 19 – FCHIDx** Fault Protection Trigger on Channel x Interrupt Enable

**Bits 0, 1, 2, 3 – CHIDx** Counter Event on Channel x Interrupt Enable

### 55.7.6 PWM Interrupt Disable Register 1

**Name:** PWM_IDR1
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FCHID3 | FCHID2 | FCHID1 | FCHID0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 16, 17, 18, 19 – FCHIDx** Fault Protection Trigger on Channel x Interrupt Disable

**Bits 0, 1, 2, 3 – CHIDx** Counter Event on Channel x Interrupt Disable

### 55.7.7 PWM Interrupt Mask Register 1

**Name:** PWM_IMR1
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FCHID3 | FCHID2 | FCHID1 | FCHID0 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 16, 17, 18, 19 – FCHIDx** Fault Protection Trigger on Channel x Interrupt Mask

**Bits 0, 1, 2, 3 – CHIDx** Counter Event on Channel x Interrupt Mask

### 55.7.8 PWM Interrupt Status Register 1

**Name:** PWM_ISR1
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

**Note:** Reading PWM_ISR1 automatically clears CHIDx and FCHIDx flags.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FCHID3 | FCHID2 | FCHID1 | FCHID0 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 16, 17, 18, 19 – FCHIDx** Fault Protection Trigger on Channel x

| Value | Description |
|---|---|
| 0 | No new trigger of the fault protection since the last read of PWM_ISR1. |
| 1 | At least one trigger of the fault protection since the last read of PWM_ISR1. |

**Bits 0, 1, 2, 3 – CHIDx** Counter Event on Channel x

| Value | Description |
|---|---|
| 0 | No new counter event has occurred since the last read of PWM_ISR1. |
| 1 | At least one counter event has occurred since the last read of PWM_ISR1. |

### 55.7.9 PWM Sync Channels Mode Register

**Name:** PWM_SCM
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS2 and WPHWS2 are cleared in the PWM Write Protection Status Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PTRCS[2:0] | | | PTRM | | | UPDM[1:0] | |
| Access | R/W | R/W | R/W | R/W | | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | SYNC3 | SYNC2 | SYNC1 | SYNC0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 23:21 – PTRCS[2:0]** DMA Controller Transfer Request Comparison Selection
Selection of the comparison used to set the flag WRDY and the corresponding DMA Controller transfer request.

**Bit 20 – PTRM** DMA Controller Transfer Request Mode

| UPDM | PTRM | WRDY Flag and DMA Controller Transfer Request |
|---|---|---|
| 0 | x | The WRDY flag in PWM Interrupt Status Register 2 and the DMA transfer request are never set to ‘1’. |
| 1 | x | The WRDY flag in PWM Interrupt Status Register 2 is set to ‘1’ as soon as the update period is elapsed, the DMA Controller transfer request is never set to ‘1’. |
| 2 | 0 | The WRDY flag in PWM Interrupt Status Register 2 and the DMA transfer request are set to ‘1’ as soon as the update period is elapsed. |
| | 1 | The WRDY flag in PWM Interrupt Status Register 2 and the DMA transfer request are set to ‘1’ as soon as the selected comparison matches. |

**Bits 17:16 – UPDM[1:0]** Synchronous Channels Update Mode

| Value | Name | Description |
|---|---|---|
| 0 | MODE0 | Manual write of double buffer registers and manual update of synchronous channels 1[(1)]. |
| 1 | MODE1 | Manual write of double buffer registers and automatic update of synchronous channels 2[(2)]. |
| 2 | MODE2 | The WRDY flag in PWM Interrupt Status Register 2 and the DMA transfer request are set to '1' as soon as the update period is elapsed. |

**Note:**

1. The update occurs at the beginning of the next PWM period, when the UPDULOCK bit in PWM Sync Channels Update Control Register is set.
2. The update occurs when the Update Period is elapsed.

**Bits 0, 1, 2, 3 – SYNCx** Synchronous Channel x

| Value | Description |
|---|---|
| 0 | Channel x is not a synchronous channel. |
| 1 | Channel x is a synchronous channel. |

### 55.7.10 PWM DMA Register

**Name:** PWM_DMAR
**Offset:** 0x24
**Reset:** –
**Property:** Write-only

Only the first 16 bits (channel counter size) are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DMADUTY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DMADUTY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DMADUTY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 23:0 – DMADUTY[23:0]** Duty-Cycle Holding Register for DMA Access
Each write access to PWM_DMAR sequentially updates PWM_CDTYUPDx.CDTYUPD with DMADUTY (only for channel configured as synchronous). See “Method 3: Automatic write of duty-cycle values and automatic trigger of the update” .

### 55.7.11 PWM Sync Channels Update Control Register

**Name:** PWM_SCUC
**Offset:** 0x28
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | UPDULOCK |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – UPDULOCK** Synchronous Channels Update Unlock
This bit is automatically reset when the update is done.

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | If the UPDM field is set to ‘0’ in PWM Sync Channels Mode Register, writing the UPDULOCK bit to ‘1’ triggers the update of the period value, the duty-cycle and the dead-time values of synchronous channels at the beginning of the next PWM period. If the field UPDM is set to ‘1’ or ‘2’, writing the UPDULOCK bit to ‘1’ triggers only the update of the period value and of the dead-time values of synchronous channels. |

### 55.7.12 PWM Sync Channels Update Period Register

**Name:** PWM_SCUP
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UPRCNT[3:0] | | | | UPR[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:4 – UPRCNT[3:0]** Update Period Counter
Reports the value of the update period counter.

**Bits 3:0 – UPR[3:0]** Update Period
Defines the time between each update of the synchronous channels if automatic trigger of the update is activated (UPDM = 1 or UPDM = 2 in PWM Sync Channels Mode Register). This time is equal to UPR+1 periods of the synchronous channels.

### 55.7.13 PWM Sync Channels Update Period Update Register

**Name:** PWM_SCUPUPD
**Offset:** 0x30
**Reset:** –
**Property:** Write-only

This register acts as a double buffer for the UPR value. This prevents an unexpected automatic trigger of the update of synchronous channels.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | UPRUPD[3:0] | | | |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 3:0 – UPRUPD[3:0]** Update Period Update
Defines the wanted time between each update of the synchronous channels if automatic trigger of the update is activated (UPDM = 1 or UPDM = 2 in PWM Sync Channels Mode Register). This time is equal to UPR+1 periods of the synchronous channels.

### 55.7.14 PWM Interrupt Enable Register 2

**Name:** PWM_IER2
**Offset:** 0x34
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CMPU7 | CMPU6 | CMPU5 | CMPU4 | CMPU3 | CMPU2 | CMPU1 | CMPU0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMPM7 | CMPM6 | CMPM5 | CMPM4 | CMPM3 | CMPM2 | CMPM1 | CMPM0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | UNRE | | | WRDY |
| Access | | | | | W | | | W |
| Reset | | | | | – | | | – |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – CMPUx** Comparison x Update Interrupt Enable

**Bits 8, 9, 10, 11, 12, 13, 14, 15 – CMPMx** Comparison x Match Interrupt Enable

**Bit 3 – UNRE** Synchronous Channels Update Underrun Error Interrupt Enable

**Bit 0 – WRDY** Write Ready for Synchronous Channels Update Interrupt Enable

### 55.7.15 PWM Interrupt Disable Register 2

**Name:** PWM_IDR2
**Offset:** 0x38
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CMPU7 | CMPU6 | CMPU5 | CMPU4 | CMPU3 | CMPU2 | CMPU1 | CMPU0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMPM7 | CMPM6 | CMPM5 | CMPM4 | CMPM3 | CMPM2 | CMPM1 | CMPM0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | UNRE | | | WRDY |
| Access | | | | | W | | | W |
| Reset | | | | | – | | | – |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – CMPUx** Comparison x Update Interrupt Disable

**Bits 8, 9, 10, 11, 12, 13, 14, 15 – CMPMx** Comparison x Match Interrupt Disable

**Bit 3 – UNRE** Synchronous Channels Update Underrun Error Interrupt Disable

**Bit 0 – WRDY** Write Ready for Synchronous Channels Update Interrupt Disable

### 55.7.16 PWM Interrupt Mask Register 2

**Name:** PWM_IMR2
**Offset:** 0x3C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CMPU7 | CMPU6 | CMPU5 | CMPU4 | CMPU3 | CMPU2 | CMPU1 | CMPU0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMPM7 | CMPM6 | CMPM5 | CMPM4 | CMPM3 | CMPM2 | CMPM1 | CMPM0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | UNRE | | | WRDY |
| Access | | | | | R | | | R |
| Reset | | | | | 0 | | | 0 |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – CMPUx** Comparison x Update Interrupt Mask

**Bits 8, 9, 10, 11, 12, 13, 14, 15 – CMPMx** Comparison x Match Interrupt Mask

**Bit 3 – UNRE** Synchronous Channels Update Underrun Error Interrupt Mask

**Bit 0 – WRDY** Write Ready for Synchronous Channels Update Interrupt Mask

### 55.7.17 PWM Interrupt Status Register 2

**Name:** PWM_ISR2
**Offset:** 0x40
**Reset:** 0x00000000
**Property:** Read-only

Reading PWM_ISR2 automatically clears flags WRDY, UNRE and CMPSx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CMPU7 | CMPU6 | CMPU5 | CMPU4 | CMPU3 | CMPU2 | CMPU1 | CMPU0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMPM7 | CMPM6 | CMPM5 | CMPM4 | CMPM3 | CMPM2 | CMPM1 | CMPM0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | UNRE | | | WRDY |
| Access | | | | | R | | | R |
| Reset | | | | | 0 | | | 0 |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – CMPUx** Comparison x Update

| Value | Description |
|---|---|
| 0 | The comparison x has not been updated since the last read of the PWM_ISR2 register. |
| 1 | The comparison x has been updated at least one time since the last read of the PWM_ISR2 register. |

**Bits 8, 9, 10, 11, 12, 13, 14, 15 – CMPMx** Comparison x Match

| Value | Description |
|---|---|
| 0 | The comparison x has not matched since the last read of the PWM_ISR2 register. |
| 1 | The comparison x has matched at least one time since the last read of the PWM_ISR2 register. |

**Bit 3 – UNRE** Synchronous Channels Update Underrun Error

| Value | Description |
|---|---|
| 0 | No Synchronous Channels Update Underrun has occurred since the last read of the PWM_ISR2 register. |
| 1 | At least one Synchronous Channels Update Underrun has occurred since the last read of the PWM_ISR2 register. |

**Bit 0 – WRDY** Write Ready for Synchronous Channels Update

| Value | Description |
|---|---|
| 0 | New duty-cycle and dead-time values for the synchronous channels cannot be written. |
| 1 | New duty-cycle and dead-time values for the synchronous channels can be written. |

### 55.7.18 PWM Output Override Value Register

**Name:** PWM_OOV
**Offset:** 0x44
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OOVL3 | OOVL2 | OOVL1 | OOVL0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OOVH3 | OOVH2 | OOVH1 | OOVH0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 16, 17, 18, 19 – OOVLx** Output Override Value for PWML output of the channel x

| Value | Description |
|---|---|
| 0 | Override value is 0 for PWML output of channel x. |
| 1 | Override value is 1 for PWML output of channel x. |

**Bits 0, 1, 2, 3 – OOVHx** Output Override Value for PWMH output of the channel x

| Value | Description |
|---|---|
| 0 | Override value is 0 for PWMH output of channel x. |
| 1 | Override value is 1 for PWMH output of channel x. |

### 55.7.19 PWM Output Selection Register

**Name:** PWM_OS
**Offset:** 0x48
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSL3 | OSL2 | OSL1 | OSL0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSH3 | OSH2 | OSH1 | OSH0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 16, 17, 18, 19 – OSLx** Output Selection for PWML output of the channel x

| Value | Description |
|---|---|
| 0 | Dead-time generator output DTOLx selected as PWML output of channel x. |
| 1 | Output override value OOVLx selected as PWML output of channel x. |

**Bits 0, 1, 2, 3 – OSHx** Output Selection for PWMH output of the channel x

| Value | Description |
|---|---|
| 0 | Dead-time generator output DTOHx selected as PWMH output of channel x. |
| 1 | Output override value OOVHx selected as PWMH output of channel x. |

### 55.7.20 PWM Output Selection Set Register

**Name:** PWM_OSS
**Offset:** 0x4C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSSL3 | OSSL2 | OSSL1 | OSSL0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSSH3 | OSSH2 | OSSH1 | OSSH0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 16, 17, 18, 19 – OSSLx** Output Selection Set for PWML output of the channel x

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Output override value OOVLx selected as PWML output of channel x. |

**Bits 0, 1, 2, 3 – OSSHx** Output Selection Set for PWMH output of the channel x

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Output override value OOVHx selected as PWMH output of channel x. |

### 55.7.21 PWM Output Selection Clear Register

**Name:** PWM_OSC
**Offset:** 0x50
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSCL3 | OSCL2 | OSCL1 | OSCL0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSCH3 | OSCH2 | OSCH1 | OSCH0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 16, 17, 18, 19 – OSCLx** Output Selection Clear for PWML output of the channel x

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Dead-time generator output DTOLx selected as PWML output of channel x. |

**Bits 0, 1, 2, 3 – OSCHx** Output Selection Clear for PWMH output of the channel x

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Dead-time generator output DTOHx selected as PWMH output of channel x. |

### 55.7.22 PWM Output Selection Set Update Register

**Name:** PWM_OSSUPD
**Offset:** 0x54
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSSUPL3 | OSSUPL2 | OSSUPL1 | OSSUPL0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSSUPH3 | OSSUPH2 | OSSUPH1 | OSSUPH0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 16, 17, 18, 19 – OSSUPLx** Output Selection Set for PWML output of the channel x

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Output override value OOVLx selected as PWML output of channel x at the beginning of the next channel x PWM period. |

**Bits 0, 1, 2, 3 – OSSUPHx** Output Selection Set for PWMH output of the channel x

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Output override value OOVHx selected as PWMH output of channel x at the beginning of the next channel x PWM period. |

### 55.7.23 PWM Output Selection Clear Update Register

**Name:** PWM_OSCUPD
**Offset:** 0x58
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSCUPL3 | OSCUPL2 | OSCUPL1 | OSCUPL0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSCUPH3 | OSCUPH2 | OSCUPH1 | OSCUPH0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 16, 17, 18, 19 – OSCUPLx** Output Selection Clear for PWML output of the channel x

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Dead-time generator output DTOLx selected as PWML output of channel x at the beginning of the next channel x PWM period. |

**Bits 0, 1, 2, 3 – OSCUPHx** Output Selection Clear for PWMH output of the channel x

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Dead-time generator output DTOHx selected as PWMH output of channel x at the beginning of the next channel x PWM period. |

### 55.7.24 PWM Fault Mode Register

**Name:** PWM_FMR
**Offset:** 0x5C
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

See Fault Inputs for details on fault generation.

> ⚠ **CAUTION** To prevent an unexpected activation of the status flag FSy in the PWM Fault Status Register, the bit FMODy can be set to '1' only if the FPOLy bit has been previously configured to its final value.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FFIL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FMOD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FPOL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – FFIL[7:0]** Fault Filtering
For each bit y of FFIL, where y is the fault input number:

0: The fault input y is not filtered.

1: The fault input y is filtered.

**Bits 15:8 – FMOD[7:0]** Fault Activation Mode
For each bit y of FMOD, where y is the fault input number:

0: The fault y is active until the fault condition is removed at the peripheral(1) level.

1: The fault y stays active until the fault condition is removed at the peripheral level(1)AND until it is cleared in the PWM Fault Clear Register.

**Note:**

1. The peripheral generating the fault.

**Bits 7:0 – FPOL[7:0]** Fault Polarity
For each bit y of FPOL, where y is the fault input number:

0: The fault y becomes active when the fault input y is at 0.

1: The fault y becomes active when the fault input y is at 1.

### 55.7.25 PWM Fault Status Register

**Name:** PWM_FSR
**Offset:** 0x60
**Reset:** 0x00000000
**Property:** Read-only

Refer to Fault Inputs for details on fault generation.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FS[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FIV[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:8 – FS[7:0]** Fault Status
For each bit y of FS, where y is the fault input number:

0: The fault y is not currently active.

1: The fault y is currently active.

**Bits 7:0 – FIV[7:0]** Fault Input Value
For each bit y of FIV, where y is the fault input number:

0: The current sampled value of the fault input y is 0 (after filtering if enabled).

1: The current sampled value of the fault input y is 1 (after filtering if enabled).

### 55.7.26 PWM Fault Clear Register

**Name:** PWM_FCR
**Offset:** 0x64
**Reset:** –
**Property:** Write-only

See Fault Inputs for details on fault generation.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FCLR[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 7:0 – FCLR[7:0]** Fault Clear
For each bit y of FCLR, where y is the fault input number:

0: No effect.

1: If bit y of FMOD field is set to ‘1’ and if the fault input y is not at the level defined by the bit y of FPOL field, the fault y is cleared and becomes inactive (FMOD and FPOL fields belong to PWM Fault Mode Register), else writing this bit to ‘1’ has no effect.

### 55.7.27 PWM Fault Protection Value Register 1

**Name:** PWM_FPV1
**Offset:** 0x68
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

See Fault Inputs for details on fault generation.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FPVL3 | FPVL2 | FPVL1 | FPVL0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FPVH3 | FPVH2 | FPVH1 | FPVH0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 16, 17, 18, 19 – FPVLx** Fault Protection Value for PWML output on channel x
This bit is taken into account only if the bit FPZLx is set to ‘0’ in PWM Fault Protection Value Register 2.

| Value | Description |
|---|---|
| 0 | PWML output of channel x is forced to ‘0’ when fault occurs. |
| 1 | PWML output of channel x is forced to ‘1’ when fault occurs. |

**Bits 0, 1, 2, 3 – FPVHx** Fault Protection Value for PWMH output on channel x
This bit is taken into account only if the bit FPZHx is set to ‘0’ in PWM Fault Protection Value Register 2.

| Value | Description |
|---|---|
| 0 | PWMH output of channel x is forced to ‘0’ when fault occurs. |
| 1 | PWMH output of channel x is forced to ‘1’ when fault occurs. |

### 55.7.28 PWM Fault Protection Enable Register

**Name:** PWM_FPE
**Offset:** 0x6C
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

Only the first 6 bits (number of fault input pins) of the register fields are significant.

Refer to Section 6.4 “Fault Inputs” for details on fault generation.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FPE3[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FPE2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FPE1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FPE0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0:7, 8:15, 16:23, 24:31 – FPEx** Fault Protection Enable for channel x
For each bit y of FPEx, where y is the fault input number:

0: Fault y is not used for the fault protection of channel x.

1: Fault y is used for the fault protection of channel x.

**⚠ CAUTION** To prevent an unexpected activation of the fault protection, the bit y of FPEx field can be set to ‘1’ only if the corresponding FPOL field has been previously configured to its final value in PWM Fault Mode Register.

### 55.7.29 PWM Event Line x Mode Register

**Name:** PWM_ELMRx
**Offset:** 0x7C + x*0x04 [x=0..1]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CSEL7 | CSEL6 | CSEL5 | CSEL4 | CSEL3 | CSEL2 | CSEL1 | CSEL0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – CSELy** Comparison y Selection

| Value | Description |
|---|---|
| 0 | A pulse is not generated on the event line x when the comparison y matches. |
| 1 | A pulse is generated on the event line x when the comparison y match. |

### 55.7.30 PWM Spread Spectrum Register

**Name:** PWM_SSPR
**Offset:** 0xA0
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS3 and WPHWS3 are cleared in the PWM Write Protection Status Register.

Only the first 16 bits (channel counter size) are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SPRDM |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SPRD[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SPRD[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SPRD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 24 – SPRDM** Spread Spectrum Counter Mode

| Value | Description |
|---|---|
| 0 | Triangular mode. The spread spectrum counter starts to count from -SPRD when the channel 0 is enabled and counts upwards at each PWM period. When it reaches +SPRD, it restarts to count from -SPRD again. |
| 1 | Random mode. The spread spectrum counter is loaded with a new random value at each PWM period. This random value is uniformly distributed and is between -SPRD and +SPRD. |

**Bits 23:0 – SPRD[23:0]** Spread Spectrum Limit Value
The spread spectrum limit value defines the range for the spread spectrum counter. It is introduced in order to achieve constant varying PWM period for the output waveform.

### 55.7.31 PWM Spread Spectrum Update Register

**Name:** PWM_SSPUP
**Offset:** 0xA4
**Reset:** –
**Property:** Write-only

This register can only be written if bits WPSWS3 and WPHWS3 are cleared in the PWM Write Protection Status Register.

This register acts as a double buffer for the SPRD value. This prevents an unexpected waveform when modifying the spread spectrum limit value.

Only the first 16 bits (channel counter size) are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SPRDUP[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SPRDUP[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SPRDUP[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 23:0 – SPRDUP[23:0]** Spread Spectrum Limit Value Update
The spread spectrum limit value defines the range for the spread spectrum counter. It is introduced in order to achieve constant varying period for the output waveform.

### 55.7.32 PWM Stepper Motor Mode Register

**Name:** PWM_SMMR
**Offset:** 0xB0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bits 16, 17 – DOWNx** Down Count

| Value | Description |
|---|---|
| 0 | Up counter. |
| 1 | Down counter. |

**Bits 0, 1 – GCENx** Gray Count Enable

| Value | Description |
|---|---|
| 0 | Disable Gray count generation on PWML[2*x], PWMH[2*x], PWML[2*x +1], PWMH[2*x +1] |
| 1 | Enable Gray count generation on PWML[2*x], PWMH[2*x], PWML[2*x +1], PWMH[2*x +1]. |

### 55.7.33 PWM Fault Protection Value Register 2

**Name:** PWM_FPV2
**Offset:** 0xC0
**Reset:** 0x000F000F
**Property:** Read/Write

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FPZL3 | FPZL2 | FPZL1 | FPZL0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 1 | 1 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FPZH3 | FPZH2 | FPZH1 | FPZH0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 1 | 1 | 1 | 1 |

**Bits 16, 17, 18, 19 – FPZLx** Fault Protection to Hi-Z for PWML output on channel x

| Value | Description |
|---|---|
| 0 | When fault occurs, PWML output of channel x is forced to value defined by the bit FPVLx in PWM Fault Protection Value Register 1. |
| 1 | When fault occurs, PWML output of channel x is forced to high-impedance state. |

**Bits 0, 1, 2, 3 – FPZHx** Fault Protection to Hi-Z for PWMH output on channel x

| Value | Description |
|---|---|
| 0 | When fault occurs, PWMH output of channel x is forced to value defined by the bit FPVHx in PWM Fault Protection Value Register 1. |
| 1 | When fault occurs, PWMH output of channel x is forced to high-impedance state. |

### 55.7.34 PWM Write Protection Control Register

**Name:** PWM_WPCR
**Offset:** 0xE4
**Reset:** –
**Property:** Write-only

See Register Write Protection for the list of registers that can be write-protected.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WPRG5 | WPRG4 | WPRG3 | WPRG2 | WPRG1 | WPRG0 | WPCMD[1:0] | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | – | 0 | – |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x50574D | PASSWD | Writing any other value in this field aborts the write operation of the WPCMD field.<br>Always reads as 0 |

**Bits 2, 3, 4, 5, 6, 7 – WPRGx** Write Protection Register Group x

| Value | Description |
|---|---|
| 0 | The WPCMD command has no effect on the register group x. |
| 1 | The WPCMD command is applied to the register group x. |

**Bits 1:0 – WPCMD[1:0]** Write Protection Command
This command is performed only if the WPKEY corresponds to 0x50574D (“PWM” in ASCII).

| Value | Name | Description |
|---|---|---|
| 0 | DISABLE_SW_PROT | Disables the software write protection of the register groups of which the bit WPRGx is at ‘1’. |
| 1 | ENABLE_SW_PROT | Enables the software write protection of the register groups of which the bit WPRGx is at ‘1’. |

| Value | Name | Description |
|---|---|---|
| 2 | ENABLE_HW_PROT | Enables the hardware write protection of the register groups of which the bit WPRGx is at ‘1’. Only a hardware reset of the PWM controller can disable the hardware write protection. Moreover, to meet security requirements, the PIO lines associated with the PWM can not be configured through the PIO interface. |

### 55.7.35 PWM Write Protection Status Register

**Name:** PWM_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | WPHWS5 | WPHWS4 | WPHWS3 | WPHWS2 | WPHWS1 | WPHWS0 |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WPVS | | WPSWS5 | WPSWS4 | WPSWS3 | WPSWS2 | WPSWS1 | WPSWS0 |
| Access | R | | R | R | R | R | R | R |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – WPVSRC[15:0]** Write Protect Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bits 8, 9, 10, 11, 12, 13 – WPHWSx** Write Protect HW Status

| Value | Description |
|---|---|
| 0 | The HW write protection x of the register group x is disabled. |
| 1 | The HW write protection x of the register group x is enabled. |

**Bit 7 – WPVS** Write Protect Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of PWM_WPSR. |
| 1 | At least one write protection violation has occurred since the last read of PWM_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

**Bits 0, 1, 2, 3, 4, 5 – WPSWSx** Write Protect SW Status

| Value | Description |
|---|---|
| 0 | The SW write protection x of the register group x is disabled. |

| Value | Description |
|---|---|
| 1 | The SW write protection x of the register group x is enabled. |

### 55.7.36 PWM Comparison x Value Register

**Name:** PWM_CMPVx
**Offset:** 0x0130 + x*0x10 [x=0..7]
**Reset:** 0x00000000
**Property:** Read/Write

Only the first 16 bits (channel counter size) of field CV are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CVM |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CV[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CV[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 24 – CVM** Comparison x Value Mode

| Value | Description |
|---|---|
| 0 | The comparison x between the counter of the channel 0 and the comparison x value is performed when this counter is incrementing. |
| 1 | The comparison x between the counter of the channel 0 and the comparison x value is performed when this counter is decrementing.<br>Note: This bit is not relevant if the counter of the channel 0 is left-aligned (CALG = 0 in PWM Channel Mode Register) |

**Bits 23:0 – CV[23:0]** Comparison x Value
Define the comparison x value to be compared with the counter of the channel 0.

### 55.7.37 PWM Comparison x Value Update Register

**Name:** PWM_CMPVUPDx
**Offset:** 0x0134 + x*0x10 [x=0..7]
**Reset:** –
**Property:** Write-only

This register acts as a double buffer for the CV and CVM values. This prevents an unexpected comparison x match.

Only the first 16 bits (channel counter size) of field CVUPD are significant.

**⚠ CAUTION** The write of the register PWM_CMPVUPDx must be followed by a write of the register PWM_CMPMUPDx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CVMUPD |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CVUPD[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CVUPD[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CVUPD[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bit 24 – CVMUPD** Comparison x Value Mode Update
**Note:** This bit is not relevant if the counter of the channel 0 is left-aligned (CALG = 0 in PWM Channel Mode Register)

| Value | Description |
|---|---|
| 0 | The comparison x between the counter of the channel 0 and the comparison x value is performed when this counter is incrementing. |
| 1 | The comparison x between the counter of the channel 0 and the comparison x value is performed when this counter is decrementing. |

**Bits 23:0 – CVUPD[23:0]** Comparison x Value Update
Define the comparison x value to be compared with the counter of the channel 0.

### 55.7.38 PWM Comparison x Mode Register

**Name:** PWM_CMPMx
**Offset:** 0x0138 + x*0x10 [x=0..7]
**Reset:** 0x00000000
**Property:** R/W

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CTR[3:0] | | | | | | | CEN |
| Access | R/W | R/W | R/W | R/W | | | | R/W |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

**Bits 23:20 – CUPRCNT[3:0]** Comparison x Update Period Counter
Reports the value of the comparison x update period counter.

Note: The field CUPRCNT is read-only

**Bits 19:16 – CUPR[3:0]** Comparison x Update Period
Defines the time between each update of the comparison x mode and the comparison x value. This time is equal to CUPR+1 periods of the channel 0 counter.

**Bits 15:12 – CPRCNT[3:0]** Comparison x Period Counter
Reports the value of the comparison x period counter.

Note: The field CPRCNT is read-only

**Bits 11:8 – CPR[3:0]** Comparison x Period
CPR defines the maximum value of the comparison x period counter (CPRCNT). The comparison x value is performed periodically once every CPR+1 periods of the channel 0 counter.

**Bits 7:4 – CTR[3:0]** Comparison x Trigger
The comparison x is performed when the value of the comparison x period counter (CPRCNT) reaches the value defined by CTR.

**Bit 0 – CEN** Comparison x Enable

| Value | Description |
|---|---|
| 0 | The comparison x is disabled and can not match. |
| 1 | The comparison x is enabled and can match. |

### 55.7.39 PWM Comparison x Mode Update Register

**Name:** PWM_CMPMUPDx
**Offset:** 0x013C + x*0x10 [x=0..7]
**Reset:** –
**Property:** W

This register acts as a double buffer for the CEN, CTR, CPR and CUPR values. This prevents an unexpected comparison x match.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CUPRUPD[3:0] | | | |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CPRUPD[3:0] | | | |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CTRUPD[3:0] | | | | | | | CENUPD |
| Access | W | W | W | W | | | | W |
| Reset | 0 | 0 | 0 | – | | | | – |

**Bits 19:16 – CUPRUPD[3:0]** Comparison x Update Period Update
Defines the time between each update of the comparison x mode and the comparison x value. This time is equal to CUPR+1 periods of the channel 0 counter.

**Bits 11:8 – CPRUPD[3:0]** Comparison x Period Update
CPR defines the maximum value of the comparison x period counter (CPRCNT). The comparison x value is performed periodically once every CPR+1 periods of the channel 0 counter.

**Bits 7:4 – CTRUPD[3:0]** Comparison x Trigger Update
The comparison x is performed when the value of the comparison x period counter (CPRCNT) reaches the value defined by CTR.

**Bit 0 – CENUPD** Comparison x Enable Update

| Value | Description |
|---|---|
| 0 | The comparison x is disabled and can not match. |
| 1 | The comparison x is enabled and can match. |

### 55.7.40 PWM Channel Mode Register

**Name:** PWM_CMRx
**Offset:** 0x0200 + x*0x20 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS2 and WPHWS2 are cleared in the PWM Write Protection Status Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PPM | DTLI | DTHI | DTE |
| Access | | | | | R/W | | | |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | TCTS | DPOLI | UPDS | CES | CPOL | CALG |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CPRE[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 19 – PPM** Push-Pull Mode
The Push-Pull mode is enabled for channel x.

| Value | Description |
|---|---|
| 0 | The Push-Pull mode is disabled for channel x. |
| 1 | The Push-Pull mode is enabled for channel x. |

**Bit 18 – DTLI** Dead-Time PWMLx Output Inverted

| Value | Description |
|---|---|
| 0 | The dead-time PWMLx output is not inverted. |
| 1 | The dead-time PWMLx output is inverted. |

**Bit 17 – DTHI** Dead-Time PWMHx Output Inverted

| Value | Description |
|---|---|
| 0 | The dead-time PWMHx output is not inverted. |
| 1 | The dead-time PWMHx output is inverted. |

**Bit 16 – DTE** Dead-Time Generator Enable

| Value | Description |
|---|---|
| 0 | The dead-time generator is disabled. |
| 1 | The dead-time generator is enabled. |

**Bit 13 – TCTS** Timer Counter Trigger Selection

| Value | Description |
|---|---|
| 0 | The comparator of the channel x (OCx) is used as the trigger source for the Timer Counter (TC). |
| 1 | The counter events of the channel x is used as the trigger source for the Timer Counter (TC). |

**Bit 12 – DPOLI** Disabled Polarity Inverted

| Value | Description |
|---|---|
| 0 | When the PWM channel x is disabled (CHIDx(PWM_SR) = 0), the OCx output waveform is the same as the one defined by the CPOL bit. |
| 1 | When the PWM channel x is disabled (CHIDx(PWM_SR) = 0), the OCx output waveform is inverted compared to the one defined by the CPOL bit. |

**Bit 11 – UPDS** Update Selection
If the PWM period is center-aligned (CALG=1):

0: The update occurs at the next end of the PWM period after writing the update register(s).

1: The update occurs at the next end of the PWM half period after writing the update register(s).

If the PWM period is left-aligned (CALG=0), the update always occurs at the end of the PWM period after writing the update register(s).

**Bit 10 – CES** Counter Event Selection
If the PWM period is center-aligned (CALG=1):

0: The channel counter event occurs at the end of the PWM period.

1: The channel counter event occurs at the end of the PWM period and at half the PWM period.

If the PWM period is left-aligned (CALG=0), the channel counter event occurs at the end of the period and the CES bit has no effect.

**Bit 9 – CPOL** Channel Polarity

| Value | Description |
|---|---|
| 0 | The OCx output waveform (output from the comparator) starts at a low level. |
| 1 | The OCx output waveform (output from the comparator) starts at a high level. |

**Bit 8 – CALG** Channel Alignment

| Value | Description |
|---|---|
| 0 | The period is left-aligned. |
| 1 | The period is center-aligned. |

**Bits 3:0 – CPRE[3:0]** Channel Prescaler

| Value | Name | Description |
|---|---|---|
| | MCK | Peripheral clock |

| Value | Name | Description |
|---|---|---|
| 1 | MCK_DIV_2 | Peripheral clock/2 |
| 2 | MCK_DIV_4 | Peripheral clock/4 |
| 3 | MCK_DIV_8 | Peripheral clock/8 |
| 4 | MCK_DIV_16 | Peripheral clock/16 |
| 5 | MCK_DIV_32 | Peripheral clock/32 |
| 6 | MCK_DIV_64 | Peripheral clock/64 |
| 7 | MCK_DIV_128 | Peripheral clock/128 |
| 8 | MCK_DIV_256 | Peripheral clock/256 |
| 9 | MCK_DIV_512 | Peripheral clock/512 |
| 10 | MCK_DIV_1024 | Peripheral clock/1024 |
| 11 | CLKA | Clock A |
| 12 | CLKB | Clock B |

### 55.7.41 PWM Channel Duty Cycle Register

**Name:** PWM_CDTYx
**Offset:** 0x0204 + x*0x20 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

Only the first 16 bits (channel counter size) are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CDTY[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CDTY[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CDTY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – CDTY[23:0]** Channel Duty-Cycle
Defines the waveform duty-cycle. This value must be defined between 0 and CPRD (PWM_CPRDx).

### 55.7.42 PWM Channel Duty Cycle Update Register

**Name:** PWM_CDTYUPDx
**Offset:** 0x0208 + x*0x20 [x=0..3]
**Reset:** –
**Property:** Write-only

This register acts as a double buffer for the CDTY value. This prevents an unexpected waveform when modifying the waveform duty-cycle.

Only the first 16 bits (channel counter size) are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CDTYUPD[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CDTYUPD[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CDTYUPD[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 23:0 – CDTYUPD[23:0]** Channel Duty-Cycle Update
Defines the waveform duty-cycle. This value must be defined between 0 and CPRD (PWM_CPRDx).

### 55.7.43 PWM Channel Period Register

**Name:** PWM_CPRDx
**Offset:** 0x020C + x*0x20 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS3 and WPHWS3 are cleared in the PWM Write Protection Status Register.

Only the first 16 bits (channel counter size) are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CPRD[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CPRD[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CPRD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – CPRD[23:0]** Channel Period
If the waveform is left-aligned, then the output waveform period depends on the channel counter source clock and can be calculated:

– By using the PWM peripheral clock divided by a given prescaler value “X” (where X = $2^{PREA}$ is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

$$\frac{(X \times \mathrm{CPRD})}{f_{\text{peripheral clock}}}$$

– By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

$$\frac{(X \times \mathrm{CPRD} \times \mathrm{DIVA})}{f_{\text{peripheral clock}}} \text{ or } \frac{(X \times \mathrm{CPRD} \times \mathrm{DIVB})}{f_{\text{peripheral clock}}}$$

If the waveform is center-aligned, then the output waveform period depends on the channel counter source clock and can be calculated:

– By using the PWM peripheral clock divided by a given prescaler value “X” (where X = $2^{PREA}$ is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

$$\frac{(2 \times X \times \mathrm{CPRD})}{f_{\text{peripheral clock}}}$$

– By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

$$\frac{(2 \times X \times \mathrm{CPRD} \times \mathrm{DIVA})}{f_{\text{peripheral clock}}} \text{ or } \frac{(2 \times X \times \mathrm{CPRD} \times \mathrm{DIVB})}{f_{\text{peripheral clock}}}$$

### 55.7.44 PWM Channel Period Update Register

**Name:** PWM_CPRDUPDx
**Offset:** 0x0210 + x*0x20 [x=0..3]
**Reset:** –
**Property:** Write-only

This register can only be written if bits WPSWS3 and WPHWS3 are cleared in the PWM Write Protection Status Register.

This register acts as a double buffer for the CPRD value. This prevents an unexpected waveform when modifying the waveform period.

Only the first 16 bits (channel counter size) are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CPRDUPD[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CPRDUPD[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CPRDUPD[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 23:0 – CPRDUPD[23:0]** Channel Period Update
If the waveform is left-aligned, then the output waveform period depends on the channel counter source clock and can be calculated:

– By using the PWM peripheral clock divided by a given prescaler value “X” (where X = $2^{PREA}$ is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

$$\frac{(X \times \mathrm{CPRDUPD})}{f_{\mathrm{peripheral\ clock}}}$$

– By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

$$\frac{(X \times \mathrm{CPRDUPD} \times \mathrm{DIVA})}{f_{\mathrm{peripheral\ clock}}} \text{ or } \frac{(X \times \mathrm{CPRDUPD} \times \mathrm{DIVB})}{f_{\mathrm{peripheral\ clock}}}$$

If the waveform is center-aligned, then the output waveform period depends on the channel counter source clock and can be calculated:

– By using the PWM peripheral clock divided by a given prescaler value “X” (where X = $2^{PREA}$ is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

$$\frac{(2 \times X \times \mathrm{CPRDUPD})}{f_{\mathrm{peripheral\ clock}}}$$

– By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

$$\frac{(2 \times X \times \mathrm{CPRDUPD} \times \mathrm{DIVA})}{f_{\mathrm{peripheral\ clock}}} \text{ or } \frac{(2 \times X \times \mathrm{CPRDUPD} \times \mathrm{DIVB})}{f_{\mathrm{peripheral\ clock}}}$$

### 55.7.45 PWM Channel Counter Register

**Name:** PWM_CCNTx
**Offset:** 0x0214 + x*0x20 [x=0..3]
**Reset:** 0x00000000
**Property:** Read-only

Only the first 16 bits (channel counter size) are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CNT[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CNT[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CNT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – CNT[23:0]** Channel Counter Register
Channel counter value. This register is reset when:

• the channel is enabled (writing CHIDx in the PWM_ENA register).

• the channel counter reaches CPRD value defined in the PWM_CPRDx register if the waveform is left-aligned.

### 55.7.46 PWM Channel Dead Time Register

**Name:** PWM_DTx
**Offset:** 0x0218 + x*0x20 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS4 and WPHWS4 are cleared in the PWM Write Protection Status Register.

Only the first 16 bits (dead-time counter size) of fields DTH and DTL are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DTL[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DTL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DTH[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DTH[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – DTL[15:0]** Dead-Time Value for PWMLx Output
Defines the dead-time value for PWMLx output. This value must be defined between 0 and CDTY (PWM_CDTYx).

**Bits 15:0 – DTH[15:0]** Dead-Time Value for PWMHx Output
Defines the dead-time value for PWMHx output. This value must be defined between 0 and the value (CPRD – CDTY) (PWM_CPRDx and PWM_CDTYx).

### 55.7.47 PWM Channel Dead Time Update Register

**Name:** PWM_DTUPDx
**Offset:** 0x021C + x*0x20 [x=0..3]
**Reset:** 0x00000000
**Property:** Write-only

This register can only be written if bits WPSWS4 and WPHWS4 are cleared in the PWM Write Protection Status Register.

This register acts as a double buffer for the DTH and DTL values. This prevents an unexpected waveform when modifying the dead-time values.

Only the first 16 bits (dead-time counter size) of fields DTHUPD and DTLUPD are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DTLUPD[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DTLUPD[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DTHUPD[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DTHUPD[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:16 – DTLUPD[15:0]** Dead-Time Value Update for PWMLx Output
Defines the dead-time value for PWMLx output. This value must be defined between 0 and CDTY (PWM_CDTYx). This value is applied only at the beginning of the next channel x PWM period.

**Bits 15:0 – DTHUPD[15:0]** Dead-Time Value Update for PWMHx Output
Defines the dead-time value for PWMHx output. This value must be defined between 0 and the value (CPRD – CDTY) (PWM_CPRDx and PWM_CDTYx). This value is applied only at the beginning of the next channel x PWM period.

### 55.7.48 PWM Channel Mode Update Register

**Name:** PWM_CMUPDx
**Offset:** 0x0400 + x*0x20 [x=0..3]
**Reset:** –
**Property:** Write-only

This register can only be written if bits WPSWS2 and WPHWS2 are cleared in the PWM Write Protection Status Register.

This register acts as a double buffer for the CPOL value. This prevents an unexpected waveform when modifying the polarity value.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | CPOLINVUP | | | | CPOLUP | |
| Access | | | W | | | | W | |
| Reset | | | – | | | | – | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 13 – CPOLINVUP** Channel Polarity Inversion Update
If this bit is written at ‘1’, the write of the bit CPOLUP is not taken into account.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The OCx output waveform (output from the comparator) is inverted. |

**Bit 9 – CPOLUP** Channel Polarity Update
The write of this bit is taken into account only if the bit CPOLINVUP is written at ‘0’ at the same time.

| Value | Description |
|---|---|
| 0 | The OCx output waveform (output from the comparator) starts at a low level. |
| 1 | The OCx output waveform (output from the comparator) starts at a high level. |

### 55.7.49 PWM External Trigger Register

**Name:** PWM_ETRGx
**Offset:** 0x042C + x*0x20 [x=0..1]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RFEN | TRGSRC | TRGFILT | TRGEDGE | | | TRGMODE[1:0] | |
| Access | R/W | R/W | R/W | R/W | | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MAXCNT[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MAXCNT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MAXCNT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – RFEN** Recoverable Fault Enable

| Value | Description |
|---|---|
| 0 | The TRGINx signal does not generate a recoverable fault. |
| 1 | The TRGINx signal generate a recoverable fault in place of the fault x input. |

**Bit 30 – TRGSRC** Trigger Source

| Value | Description |
|---|---|
| 0 | The TRGINx signal is driven by the PWMEXTRGx input. |
| 1 | The TRGINx signal is driven by the Analog Comparator Controller. |

**Bit 29 – TRGFILT** Filtered input

| Value | Description |
|---|---|
| 0 | The external trigger input x is not filtered. |
| 1 | The external trigger input x is filtered. |

**Bit 28 – TRGEDGE** Edge Selection

| Value | Name | Description |
|---|---|---|
| 0 | FALLING_ZERO | TRGMODE = 1: TRGINx event detection on falling edge.<br>TRGMODE = 2, 3: TRGINx active level is 0 |
| 1 | RISING_ONE | TRGMODE = 1: TRGINx event detection on rising edge.<br>TRGMODE = 2, 3: TRGINx active level is 1 |

**Bits 25:24 – TRGMODE[1:0]** External Trigger Mode

| Value | Name | Description |
|---|---|---|
| 0 | OFF | External trigger is not enabled. |
| 1 | MODE1 | External PWM Reset Mode |
| 2 | MODE2 | External PWM Start Mode |
| 3 | MODE3 | Cycle-by-cycle Duty Mode |

**Bits 23:0 – MAXCNT[23:0]** Maximum Counter value
Maximum channel x counter value measured at the TRGINx event since the last read of the register.

At the TRGINx event, if the channel x counter value is greater than the stored MAXCNT value, then MAXCNT is updated by the channel x counter value.

### 55.7.50 PWM Leading-Edge Blanking Register

**Name:** PWM_LEBRx
**Offset:** 0x0430 + x*0x20 [x=0..1]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PWMHREN | PWMHFEN | PWMLREN | PWMLFEN |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | LEBDELAY[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 19 – PWMHREN** PWMH Rising Edge Enable
Leading-edge blanking is enabled on PWMHx output rising edge.

| Value | Description |
|---|---|
| 0 | Leading-edge blanking is disabled on PWMHx output rising edge. |
| 1 | Leading-edge blanking is enabled on PWMHx output rising edge. |

**Bit 18 – PWMHFEN** PWMH Falling Edge Enable

| Value | Description |
|---|---|
| 0 | Leading-edge blanking is disabled on PWMHx output falling edge. |
| 1 | Leading-edge blanking is enabled on PWMHx output falling edge. |

**Bit 17 – PWMLREN** PWML Rising Edge Enable

| Value | Description |
|---|---|
| 0 | Leading-edge blanking is disabled on PWMLx output rising edge. |
| 1 | Leading-edge blanking is enabled on PWMLx output rising edge. |

**Bit 16 – PWMLFEN** PWML Falling Edge Enable

| Value | Description |
|---|---|
| 0 | Leading-edge blanking is disabled on PWMLx output falling edge. |

| Value | Description |
|---|---|
| 1 | Leading-edge blanking is enabled on PWMLx output falling edge. |

**Bits 6:0 – LEBDELAY[6:0]** Leading-Edge Blanking Delay for TRGINx
Leading-edge blanking duration for external trigger x input. The delay is calculated according to the following formula:

LEBDELAY = ($f_{peripheral\ clock}$ × Delay) + 1

# 56. Secure Fuse Controller (SFC)

## 56.1 Description

The Secure Fuse Controller (SFC) interfaces the system with electrical fuses in a secure way.

The default value of a fuse is logic ‘0’ (not programmed). A programmed fuse is logic ‘1’.

An electrical fuse matrix is a type of non-volatile memory. Each fuse in the matrix can be programmed only one time. They are typically used to store calibration bits for analog cells such as oscillators, configuration settings, chip identifiers or cryptographic keys.

A specific number of fuse bits are programmed during the production tests through the test interface. The remaining 544 fuse bits are programmed by the user and by software through the user interface.

The SFC automatically reads the fuse values on start-up and stores them in 32-bit registers in order to make them accessible by the software. Only fuses set to level ‘1’ are programmed.

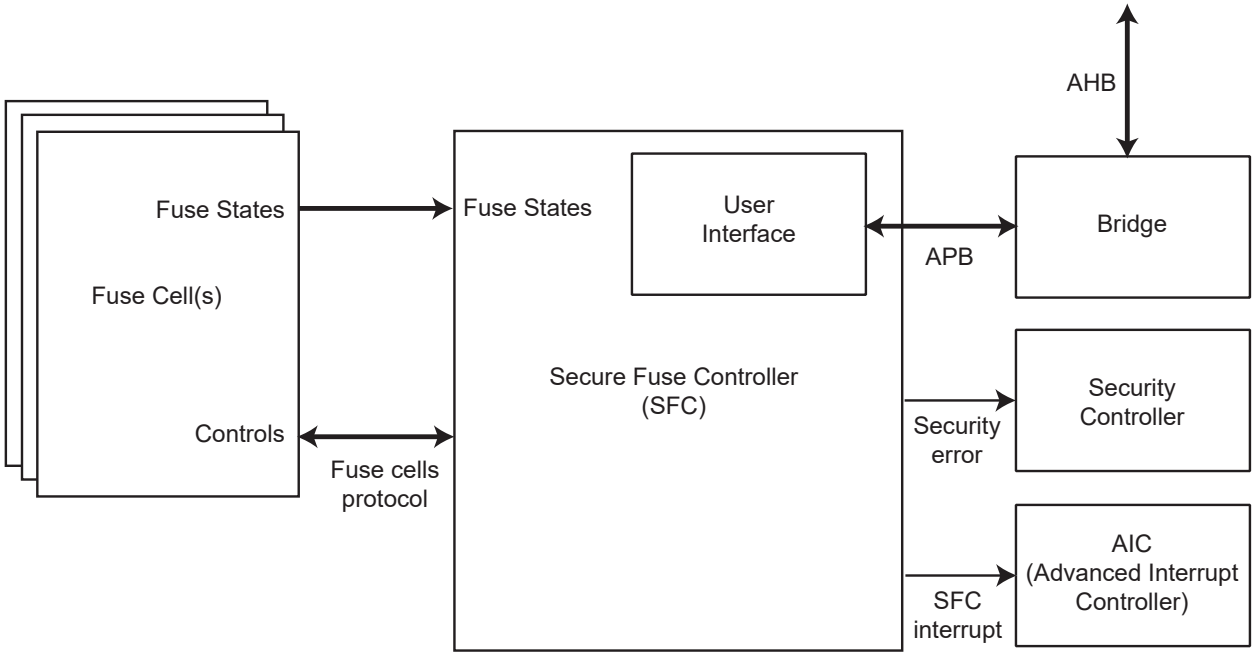
Several security mechanisms make irregular data recovery more complex to achieve.

## 56.2 Embedded Characteristics

- Fuse bits partitioned into two areas:
  - Reserved area
  - 544-bit user area
- Program and read the fuse states by software
- Automatic check of programmed fuses
- Detection of irregular alteration of the fuse states in reserved area during startup and report
- Live detection of irregular alteration of all the fuse states and report
- Part of fuse states maskable for reading

## 56.3 Block Diagram

**Figure 56-1. SFC Block Diagram**




## 56.4 Functional Description

### 56.4.1 Accessing the SFC

Setting the write-once FUSE bit in the SFR_SECURE register disables access to the Secure Fuse Controller (SFC).

### 56.4.2 Fuse Partitioning

The fuses are split into a user area of 544 bits and a reserved area.

The reserved area is typically used to store calibration bits for analog cells such as oscillators, configuration settings, chip identifiers, etc. The user area fuses are programmed later on by the user.

### 56.4.3 Fuse Integrity Checking

The SFC automatically reads the fuses values at start-up and stores them in 32-bit registers in order to make them accessible by software. At this time, the SFC checks the integrity of the fuse states in the reserved area.

If an inconsistency is detected, the CHECK error flag (named ACE for the reserved area) in the Status Register (SFC_SR) is set to ‘1’ and can trigger an interrupt. This flag is automatically cleared at ‘0’, when the Status Register (SFC_SR) is read.

### 56.4.4 Fuse Integrity Live Checking

The SFC automatically checks the integrity of all fuse states at every time after the start-up is finished. This ensures that the fuses states cannot be changed without notice.

If an inconsistency is detected, the LCHECK error flag in the Status Register (SFC_SR) is set to ‘1’ and can trigger an interrupt. This flag is automatically cleared at ‘0’, when the Status Register (SFC_SR) is read.

### 56.4.5 Fuse Access

#### 56.4.5.1 Fuse Reading

The fuse states are automatically latched at core start-up and are available for reading in the Data Registers (SFC_DRx).

The fuse states of bits 0 to 31 are available in the Data Register 0 (SFC_DR0), the fuse states of bits 32 to 63 are available in the Data Register 1 (SFC_DR1) and so on.

When fuse programming is performed, the fuse states are automatically updated in the Data Registers (SFC_DRx).

#### 56.4.5.2 Fuse Programming

All the fuses can be written by software.

The sequence of instructions to program fuses is the following:

1. Write the key code 0xFB in the Key Register (SFC_KR).
2. Write the word to program in the corresponding Data Register (SFC_DRx).
   For example, if fuses 0 to 31 must be programmed, Data Register 0 (SFC_DR0) must be written. If fuses 32 to 61 must be programmed, Data Register 1 (SFC_DR1) must be written. Only the data bits set to level ‘1’ are programmed.
3. Wait for flag PGMC to rise in the Status Register (SFC_SR) by polling or interrupt.
4. Check the value of flag PGMF: if it is set to 1, it means that the programming procedure failed. After programming, the fuses are read back in the corresponding SFC_DRx.

#### 56.4.5.3 Fuse Masking

It is possible to mask a fuse array. Once the fuse masking is enabled, the data registers from SFC_DR0 to SFC_DR7 are read at a value of ‘0’, regardless of the fuse state (the registers that are masked depend on the SFC hardware customizing).

To activate fuse masking, the MSK bit of the SFC Mode Register (SFC_MR) must be written to level ‘1’. The MSK bit is set-only. Only a hardware reset can disable fuse masking.

The MSK bit has no effect on the programming of masked fuses.

### 56.4.6 Fuse Functions

The “Fuse Box Controller” section defines the fuse bits that can be used as general purpose bits when standard boot is used.

If secure boot is used, refer to the device “Secure Boot Strategy” application note included in the Secure Package.

## 56.5 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | SFC_KR | 7:0 | KEY[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | SFC_MR | 7:0 | | | | SASEL | | | | MSK |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | SFC_IER | 7:0 | | | | LCHECK | | | PGMF | PGMC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | ACE | |
| | | 31:24 | | | | | | | | |
| 0x14 | SFC_IDR | 7:0 | | | | LCHECK | | | PGMF | PGMC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | ACE | |
| | | 31:24 | | | | | | | | |
| 0x18 | SFC_IMR | 7:0 | | | | LCHECK | | | PGMF | PGMC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | ACE | |
| | | 31:24 | | | | | | | | |
| 0x1C | SFC_SR | 7:0 | | | | LCHECK | | | PGMF | PGMC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | ACE | APLE |
| | | 31:24 | | | | | | | | |
| 0x20 | SFC_DR0 | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | DATA[23:16] | | | | | | | |
| | | 31:24 | DATA[31:24] | | | | | | | |
| ... | | | | | | | | | | |
| 0x60 | SFC_DR16 | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | DATA[23:16] | | | | | | | |
| | | 31:24 | DATA[31:24] | | | | | | | |

### 56.5.1 SFC Key Register

**Name:** SFC_KR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | KEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 7:0 – KEY[7:0]** Key Code
This field must be written with the correct key code (0xFB) prior to any write in a Data Register (SFC_DRx) in order to enable the fuse programming. For each write of SFC_DRx, this field must be written immediately before.

### 56.5.2 SFC Mode Register

**Name:** SFC_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | SASEL | | | | MSK |
| Access | | | | R/W | | | | R/W |
| Reset | | | | 0 | | | | 0 |

**Bit 4 – SASEL** Sense Amplifier Selection

| Value | Description |
|---|---|
| 0 | Comparator type sense amplifier selected |
| 1 | Latch type sense amplifier selected |

**Bit 0 – MSK** Mask Data Registers
The MSK bit is set-only. Only a hardware reset can disable fuse masking.

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | The data registers from SFC_DR0 to SFC_DR7 are always read at 0x00000000. |

### 56.5.3 SFC Interrupt Enable Register

**Name:** SFC_IER
**Offset:** 0x10
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ACE | |
| Access | | | | | | | W | |
| Reset | | | | | | | – | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | LCHECK | | | PGMF | PGMC |
| Access | | | | W | | | W | W |
| Reset | | | | – | | | – | – |

**Bit 17 – ACE** Manufacturer Area Check Error Interrupt Enable

**Bit 4 – LCHECK** Live Integrity Check Error Interrupt Enable

**Bit 1 – PGMF** Programming Sequence Failed Interrupt Enable

**Bit 0 – PGMC** Programming Sequence Completed Interrupt Enable

### 56.5.4 SFC Interrupt Disable Register

**Name:** SFC_IDR
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ACE | |
| Access | | | | | | | W | |
| Reset | | | | | | | – | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | LCHECK | | | PGMF | PGMC |
| Access | | | | W | | | W | W |
| Reset | | | | – | | | – | – |

**Bit 17 – ACE** Manufacturer Area Check Error Interrupt Disable

**Bit 4 – LCHECK** Live Integrity Check Error Interrupt Disable

**Bit 1 – PGMF** Programming Sequence Failed Interrupt Disable

**Bit 0 – PGMC** Programming Sequence Completed Interrupt Disable

### 56.5.5 SFC Interrupt Mask Register

**Name:** SFC_IMR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: Corresponding interrupt is not enabled.

1: Corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ACE | |
| Access | | | | | | | R | |
| Reset | | | | | | | 0 | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | LCHECK | | | PGMF | PGMC |
| Access | | | | R | | | R | R |
| Reset | | | | 0 | | | 0 | 0 |

**Bit 17 – ACE** Manufacturer Area Check Error Interrupt Mask

**Bit 4 – LCHECK** Live Integrity Checking Error Interrupt Mask

**Bit 1 – PGMF** Programming Sequence Failed Interrupt Mask

**Bit 0 – PGMC** Programming Sequence Completed Interrupt Mask

### 56.5.6 SFC Status Register

**Name:** SFC_SR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ACE | APLE |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | LCHECK | | | PGMF | PGMC |
| Access | | | | R | | | R | R |
| Reset | | | | 0 | | | 0 | 0 |

**Bit 17 – ACE** Manufacturer Area Check Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No check error in the reserved area since the last read of SFC_SR. |
| 1 | At least one check error in the reserved area since the last read of SFC_SR. |

**Bit 16 – APLE** Area Programming Lock Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No programming attempt has been made in the locked area since the last read of SFC_SR. |
| 1 | A programming attempt has been made in the locked area since the last read of SFC_SR. |

**Bit 4 – LCHECK** Live Integrity Checking Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No live integrity check error since the last read of SFC_SR. |
| 1 | At least one live integrity check error since the last read of SFC_SR. |

**Bit 1 – PGMF** Programming Sequence Failed (cleared on read)

| Value | Description |
|---|---|
| 0 | No programming failure occurred during last programming sequence since the last read of SFC_SR. |

| Value | Description |
|---|---|
| 1 | A programming failure occurred since the last read of SFC_SR. |

**Bit 0 – PGMC** Programming Sequence Completed (cleared on read)

| Value | Description |
|---|---|
| 0 | No programming sequence completion since the last read of SFC_SR. |
| 1 | At least one programming sequence completion since the last read of SFC_SR. |

### 56.5.7 SFC Data Register x

**Name:** SFC_DRx
**Offset:** 0x20 + x*0x04 [x=0..16]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATA[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DATA[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DATA[31:0]** Fuse Data

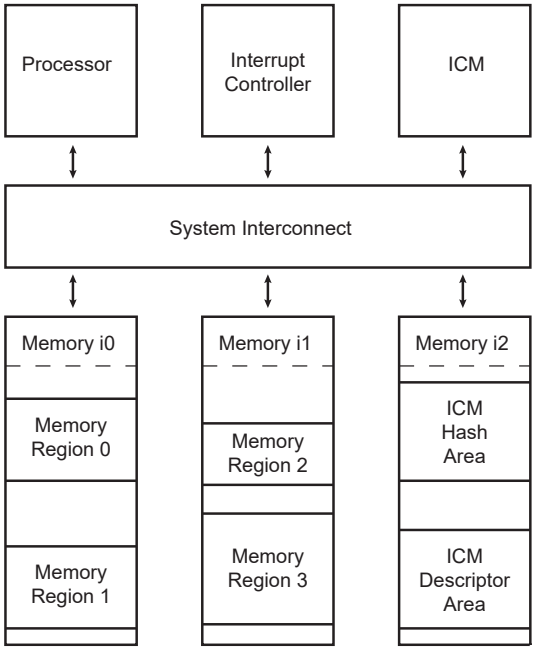
| Value | Description |
|---|---|
| READ | Reports the state of the corresponding fuses. |
| WRITE | The data to be programmed in the corresponding fuses. Only bits with a value of ‘1’ are programmed. Writing this register automatically triggers a programming sequence of the corresponding fuses. Note that a write to the Key Register (SFC_KR) with the correct key code must always precede any write to SFC_DRx. |

# 57. Integrity Check Monitor (ICM)

## 57.1 Description

The Integrity Check Monitor (ICM) is a DMA controller that performs hash calculation over multiple memory regions through the use of transfer descriptors located in memory (ICM Descriptor Area). The Hash function is based on the Secure Hash Algorithm (SHA). The ICM integrates two modes of operation. The first one is used to hash a list of memory regions and save the digests to memory (ICM Hash Area). The second mode is an active monitoring of the memory. In that mode, the hash function is evaluated and compared to the digest located at a predefined memory address (ICM Hash Area). If a mismatch occurs, an interrupt is raised. See the figure below for an example of four-region monitoring. Hash and Descriptor areas are located in Memory instance i2, and the four regions are split in memory instances i0 and i1.

**Figure 57-1. Four-region Monitoring Example**




The ICM SHA engine is compliant with the American FIPS (Federal Information Processing Standard) Publication 180-2 specification.

The following terms are concise definitions of the ICM concepts used throughout this document:

- Region—a partition of instruction or data memory space
- Region Descriptor—a data structure stored in memory, defining region attributes
- Region Attributes—region start address, region size, region SHA engine processing mode, Write Back or Compare function mode
- Context Registers—a set of ICM non-memory-mapped, internal registers which are automatically loaded, containing the attributes of the region being processed
- Main List—a list of region descriptors. Each element associates the start address of a region with a set of attributes.
- Secondary List—a linked list defined on a per region basis that describes the memory layout of the region (when the region is non-contiguous)

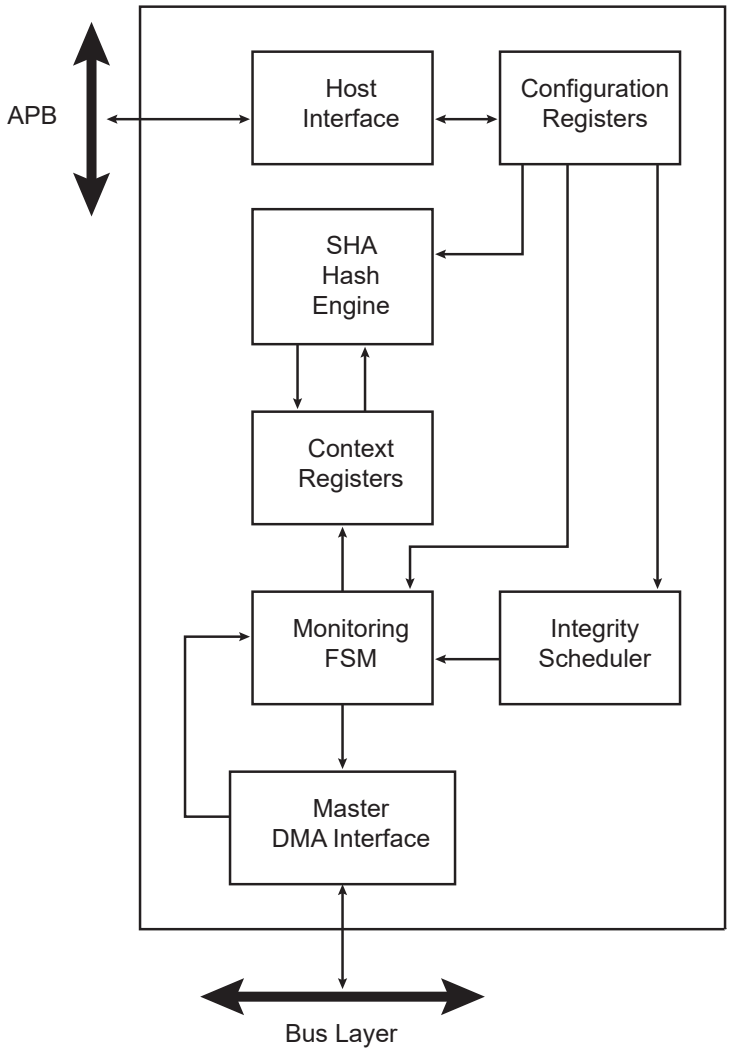
- Hash Area—predefined memory space where the region hash results (digest) are stored

## 57.2 Embedded Characteristics

- DMA AHB Master Interface
- Supports Monitoring of up to 4 Non-Contiguous Memory Regions
- Supports Block Gathering Using Linked Lists
- Supports Secure Hash Algorithm (SHA1, SHA224, SHA256)
- Compliant with FIPS Publication 180-2
- Configurable Processing Period:
  - When SHA1 algorithm is processed, the runtime period is either 85 or 209 clock cycles.
  - When SHA256 or SHA224 algorithm is processed, the runtime period is either 72 or 194 clock cycles.
- Programmable Bus Burden

## 57.3 Block Diagram

**Figure 57-2. Integrity Check Monitor Block Diagram**




## 57.4 Product Dependencies

### 57.4.1 Power Management

The peripheral clock is not continuously provided to the ICM. The programmer must first enable the ICM clock in the Power Management Controller (PMC) before using the ICM.

### 57.4.2 Interrupt Sources

The ICM interface has an interrupt line connected to the Interrupt Controller.

Handling the ICM interrupt requires programming the interrupt controller before configuring the ICM.

## 57.5 Functional Description

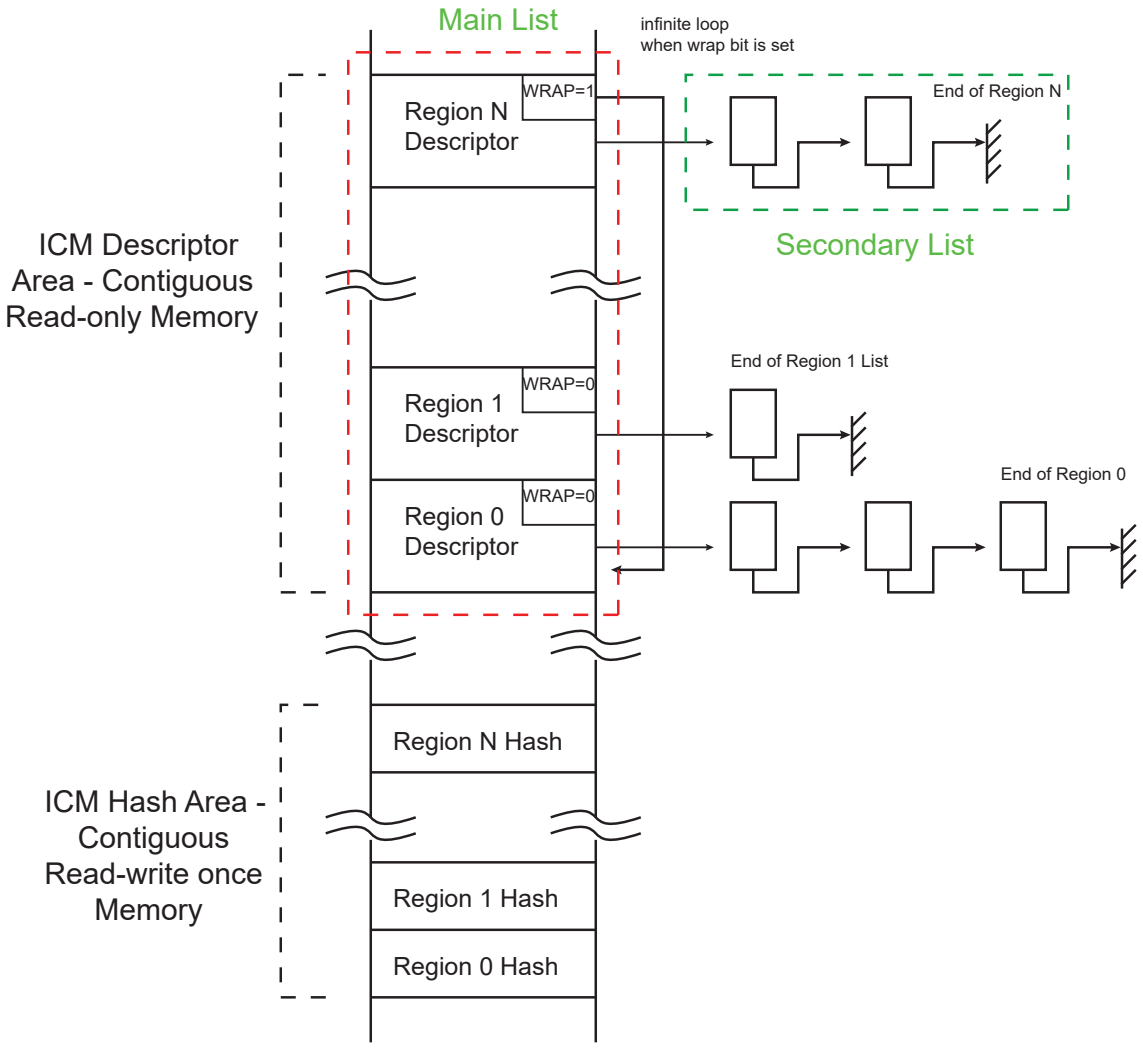
### 57.5.1 Overview

The Integrity Check Monitor (ICM) is a DMA controller that performs SHA-based memory hashing over memory regions. As shown in figure Integrity Check Monitor Block Diagram, it integrates a DMA interface, a Monitoring Finite State Machine (FSM), an integrity scheduler, a set of context registers, a SHA engine, an interface for configuration and status registers.

The ICM integrates a Secure Hash Algorithm engine (SHA). This engine requires a message padded according to FIPS180-2 specification when used as a SHA calculation unit only. Otherwise, if the ICM is used as integrated check for memory content, the padding is not mandatory. The SHA module produces an N-bit message digest each time a block is read and a processing period ends. N is 160 for SHA1, 224 for SHA224, 256 for SHA256.

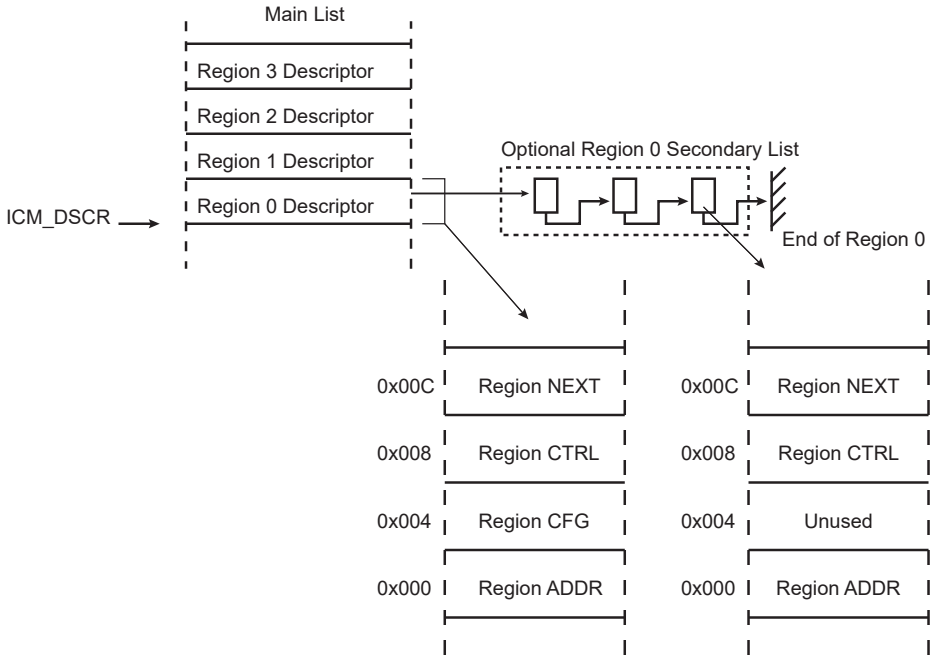
When the ICM module is enabled, it sequentially retrieves a circular list of region descriptors from the memory (Main List described in figure ICM Region Descriptor and Hash Areas). Up to four regions may be monitored. Each region descriptor is composed of four words indicating the layout of the memory region (see figure Region Descriptor ). It also contains the hashing engine configuration on a per-region basis. As soon as the descriptor is loaded from the memory and context registers are updated with the data structure, the hashing operation starts. A programmable number of blocks (see TRSIZE field of the ICM_RCTRL structure member) is transferred from the memory to the SHA engine. When the desired number of blocks have been transferred, the digest is either moved to memory (Write Back function) or compared with a digest reference located in the system memory (Compare function). If a digest mismatch occurs, an interrupt is triggered if unmasked. The ICM module passes through the region descriptor list until the end of the list marked by an end of list marker (WRAP or EOM bit in ICM_RCFG structure member set to one). To continuously monitor the list of regions, the WRAP bit must be set to one in the last data structure and EOM must be cleared.

**Figure 57-3. ICM Region Descriptor and Hash Areas**



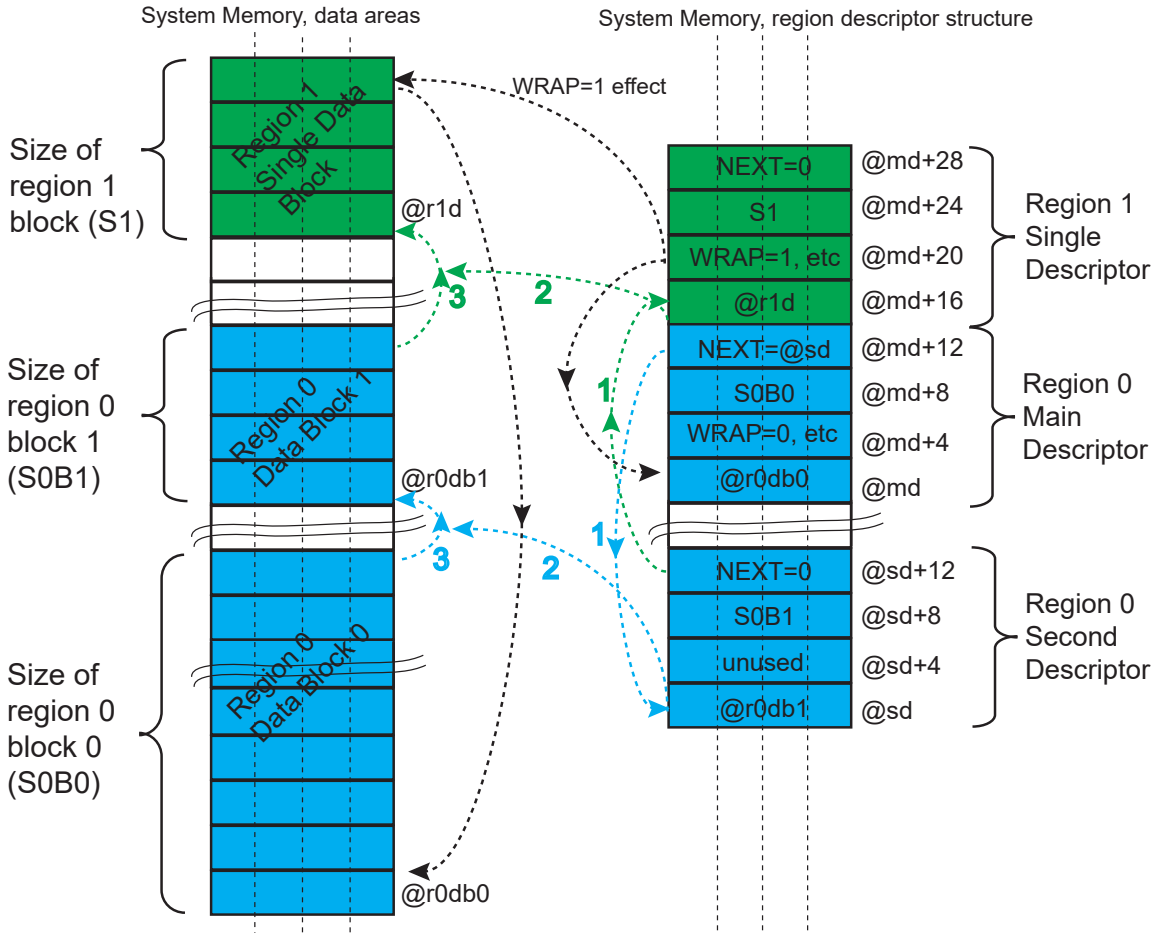

Each region descriptor supports gathering of data through the use of the Secondary List. Unlike the Main List, the Secondary List cannot modify the configuration attributes of the region. When the end of the Secondary List has been encountered, the ICM returns to the Main List. Memory integrity monitoring can be considered as a background service and the mandatory bandwidth shall be very limited. In order to limit the ICM memory bandwidth, use ICM_CFG.BBC to control the ICM memory load.

**Figure 57-4. Region Descriptor**




The figure below shows an example of the mandatory ICM settings required to monitor three memory data blocks of the system memory (defined as two regions) with one region being not contiguous (two separate areas) and one contiguous memory area. For each region, the SHA algorithm may be independently selected (different for each region). The wrap allows continuous monitoring.

**Figure 57-5. Example: Monitoring of 3 Memory Data Blocks (Defined as 2 Regions)**




### 57.5.2 ICM Region Descriptor Structure

The ICM Region Descriptor Area is a contiguous area of system memory that the controller and the processor can access. When the ICM is activated, the controller performs a descriptor fetch operation at *(ICM_DSCR) address. If the Main List contains more than one descriptor (i.e., more than one region is to be monitored), the fetch address is *(ICM_DSCR) + (RID<<4) where RID is the region identifier.

**Table 57-1. Region Descriptor Structure (Main List)**

| Offset | Structure Member | Name |
|---|---|---|
| ICM_DSCR+0x000+RID*(0x10) | ICM Region Start Address | ICM_RADDR |
| ICM_DSCR+0x004+RID*(0x10) | ICM Region Configuration | ICM_RCFG |
| ICM_DSCR+0x008+RID*(0x10) | ICM Region Control | ICM_RCTRL |
| ICM_DSCR+0x00C+RID*(0x10) | ICM Region Next Address | ICM_RNEXT |

#### 57.5.2.1 ICM Region Start Address Structure Member

**Name:** ICM_RADDR
**Property:** Read/Write

Register offset is calculated as ICM_DSCR+0x000+RID*(0x10).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

**Bits 31:0 – RADDR[31:0]** Region Start Address
This field indicates the first byte address of the region.

#### 57.5.2.2 ICM Region Configuration Structure Member

**Name:** ICM_RCFG
**Property:** Read/Write

Register offset is calculated as ICM_DSCR+0x004+RID*(0x10).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | ALGO[2:0] | | | | PROCDLY | SUIEN | ECIEN |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WCIEN | BEIEN | DMIEN | RHIEN | | EOM | WRAP | CDWBN |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | | | | | | |

**Bits 14:12 – ALGO[2:0]** SHA Algorithm
Values which are not listed in the table must be considered as “reserved”.

| Value | Name | Description |
|---|---|---|
| 0 | SHA1 | SHA1 algorithm processed |
| 1 | SHA256 | SHA256 algorithm processed |

**Bit 10 – PROCDLY** Processing Delay
When SHA1 algorithm is processed, the runtime period is either 85 or 209 clock cycles.
When SHA256 algorithm is processed, the runtime period is either 72 or 194 clock cycles.

| Value | Name | Description |
|---|---|---|
| 0 | SHORTEST | SHA processing runtime is the shortest one. |
| 1 | LONGEST | SHA processing runtime is the longest one. |

**Bit 9 – SUIEN** Monitoring Status Updated Condition Interrupt (Default Enabled)

| Value | Name | Description |
|---|---|---|
| 0 | | The ICM_ISR.RSU[i] flag is set when the corresponding descriptor is loaded from memory to ICM. |
| 1 | | The ICM_ISR.RSU[i] flag remains cleared even if the setting condition is met. |

**Bit 8 – ECIEN** End Bit Condition Interrupt (Default Enabled)

| Value | Name | Description |
|---|---|---|
| 0 | | The ICM_ISR.REC[i] flag is set when the descriptor with the EOM bit set is processed. |
| 1 | | The ICM_ISR.REC[i] flag remains cleared even if the setting condition is met. |

**Bit 7 – WCIEN** Wrap Condition Interrupt Disable (Default Enabled)

| Value | Name | Description |
|---|---|---|
| 0 | | The ICM_ISR.RWC[i] flag is set when the WRAP bit is set in a descriptor of the main list. |
| 1 | | ICM_ISR.RWC[i] flag remains cleared even if the setting condition is met. |

**Bit 6 – BEIEN** Bus Error Interrupt Disable (Default Enabled)

| Value | Name | Description |
|---|---|---|
| 0 | | The flag is set when an error is reported on the system bus by the bus matrix. |
| 1 | | The flag remains cleared even if the setting condition is met. |

**Bit 5 – DMIEN** Digest Mismatch Interrupt Disable (Default Enabled)

| Value | Name | Description |
|---|---|---|
| 0 | | The ICM_ISR.RBE[i] flag is set when the hash value just calculated from the processed region differs from expected hash value. |
| 1 | | The ICM_ISR.RBE[i] flag remains cleared even if the setting condition is met. |

**Bit 4 – RHIEN** Region Hash Completed Interrupt Disable (Default Enabled)

| Value | Name | Description |
|---|---|---|
| 0 | | The ICM_ISR.RHC[i] flag is set when the field NEXT = 0 in a descriptor of the main or second list. |
| 1 | | The ICM_ISR.RHC[i] flag remains cleared even if the setting condition is met. |

**Bit 2 – EOM** End Of Monitoring

| Value | Name | Description |
|---|---|---|
| 0 | | The current descriptor does not terminate the monitoring. |
| 1 | | The current descriptor terminates the Main List. WRAP value has no effect. |

**Bit 1 – WRAP** Wrap Command

| Value | Name | Description |
|---|---|---|
| 0 | | The next region descriptor address loaded is the current region identifier descriptor address incremented by 0x10. |
| 1 | | The next region descriptor address loaded is ICM_DSCR. |

**Bit 0 – CDWBN** Compare Digest or Write Back Digest

| Value | Name | Description |
|---|---|---|
| 0 | | The digest is written to the Hash area. |
| 1 | | The digest value is compared to the digest stored in the Hash area. |

#### 57.5.2.3 ICM Region Control Structure Member

**Name:** ICM_RCTRL
**Property:** Read/Write

Register offset is calculated as ICM_DSCR+0x008+RID*(0x10).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TRSIZE[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TRSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

**Bits 15:0 – TRSIZE[15:0]** Transfer Size for the Current Chunk of Data

#### 57.5.2.4 ICM Region Next Address Structure Member

**Name:** ICM_RNEXT
**Property:** Read/Write

Register offset is calculated as ICM_DSCR+0x00C+RID*(0x10).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[28:21] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[20:13] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[12:5] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[4:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | | | |
| Reset | | | | | | | | |

**Bits 31:3 – NEXT[28:0]** Region Transfer Descriptor Next Address
When configured to 0, this field indicates that the current descriptor is the last descriptor of the Secondary List, otherwise it points at a new descriptor of the Secondary List.

### 57.5.3 ICM Hash Area

The ICM Hash Area is a contiguous area of system memory that the controller and the processor can access. The physical location is configured in the ICM hash area start address register. This address is a multiple of 128 bytes. If the CDWBN bit of the context register is cleared (i.e., Write Back activated), the ICM performs a digest write operation at the following starting location: *(ICM_HASH) + (RID<<5), where RID is the current region context identifier. If the CDWBN bit of the context register is set (i.e., Digest Comparison activated), the ICM performs a digest read operation at the same address.

#### 57.5.3.1 Message Digest Example

Considering the following 512-bit message (example given in FIPS 180-2):

“61626380000000000

**Table 57-2. 512 bits Message Memory Mapping**

| Memory Address | Address Offset / Byte Lane | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 0x7 / 63:56 | 0x6 / 55:48 | 0x5 / 47:40 | 0x4 / 39:32 | 0x3 / 31:24 | 0x2 / 23:16 | 0x1 / 15:8 | 0x0 / 7:0 |
| 0x000 | 00 | 00 | 00 | 00 | 80 | 63 | 62 | 61 |
| 0x008–0x030 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x038 | 18 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |

The digest is stored at the memory location pointed at by the ICM_HASH pointer with a Region Offset.

**Table 57-3. LE Resulting SHA-160 Message Digest Memory Mapping**

| Memory Address | Address Offset / Byte Lane | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 0x7 / 63:56 | 0x6 / 55:48 | 0x5 / 47:40 | 0x4 / 39:32 | 0x3 / 31:24 | 0x2 / 23:16 | 0x1 / 15:8 | 0x0 / 7:0 |
| 0x000 | 6a | 81 | 06 | 47 | 36 | 3e | 99 | a9 |
| 0x008 | 6c | c2 | 50 | 78 | 71 | 25 | 3e | ba |
| 0x010 | – | – | – | – | 9d | d8 | d0 | 9c |

**Table 57-4. Resulting SHA-256 Message Digest Memory Mapping**

| Memory Address | Address Offset / Byte Lane | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 0x7 / 63:56 | 0x6 / 55:48 | 0x5 / 47:40 | 0x4 / 39:32 | 0x3 / 31:24 | 0x2 / 23:16 | 0x1 / 15:8 | 0x0 / 7:0 |
| 0x000 | ea | cf | 01 | 8f | bf | 16 | 78 | ba |
| 0x008 | 23 | 22 | ae | 5d | de | 40 | 41 | 41 |
| 0x010 | 9c | 7a | 17 | 96 | a3 | 61 | 03 | b0 |
| 0x018 | ad | 15 | 00 | f2 | 61 | ff | 10 | b4 |

Considering the following 1024-bit message (example given in FIPS 180-2):

“61626380000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000018”

The message is written to memory in a Little Endian (LE) system architecture.

**Table 57-5. 1024 bits Message Memory Mapping**

| Memory Address | Address Offset / Byte Lane | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 0x7 / 63:56 | 0x6 / 55:48 | 0x5 / 47:40 | 0x4 / 39:32 | 0x3 / 31:24 | 0x2 / 23:16 | 0x1 / 15:8 | 0x0 / 7:0 |
| 0x000 | 00 | 00 | 00 | 00 | 80 | 63 | 62 | 61 |

.....continued

| Memory Address | Address Offset / Byte Lane | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 0x7 / 63:56 | 0x6 / 55:48 | 0x5 / 47:40 | 0x4 / 39:32 | 0x3 / 31:24 | 0x2 / 23:16 | 0x1 / 15:8 | 0x0 / 7:0 |
| 0x008–0x070 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x078 | 18 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |

### 57.5.4 Using ICM as SHA Engine

The ICM can be configured to only calculate a SHA1, SHA224, SHA256 digest value.

#### 57.5.4.1 Settings for Simple SHA Calculation

The start address of the system memory containing the data to hash must be configured in the transfer descriptor of the DMA embedded in the ICM.

The transfer descriptor is a system memory area integer multiple of 4 x 32-bit words and the start address of the descriptor must be configured in ICM_DSCR (the start address must be aligned on 64-bytes; six LSB must be cleared). If the data to hash is already padded according to SHA standards, only a single descriptor is required, and ICM_RCFG.EOM must be written to 1. If the data to hash does not contain a padding area, it is possible to define the padding area in another system memory location, the ICM can be configured to automatically jump from a memory area to another one by configuring the descriptor register ICM_RNEXT with a value that differs from 0. Configuring ICM_RNEXT.NEXT with the start address of the padding area forces the ICM to concatenate both areas, thus providing the SHA result from the start address of the hash area configured in ICM_HASH.

Whether the system memory is configured as a single or multiple data block area, ICM_RCFG.CDWBN and ICM_RCFG.WRAP must be cleared. The bits WBDIS, EOMDIS, SLBDIS must be cleared in ICM_CFG.

ICM_RCTRL.RHIEN and ICM_RCTRL.ECIEN must be written to 1. The flag RHC[i], i being the region index, is set (if RHIEN is set) when the hash result is available at address defined in ICM_HASH. The flag REC[i], i being the region index, is set (if ECIEN is set) when the hash result is available at the address defined in ICM_HASH.

An interrupt is generated if the bit RHC[i] is written to 1 in the ICM_IER (if RHC[i] is set in ICM_RCTRL of region i) or if the bit REC[i] is written to 1 in the ICM_IER (if REC[i] is set in ICM_RCTRL of region i).

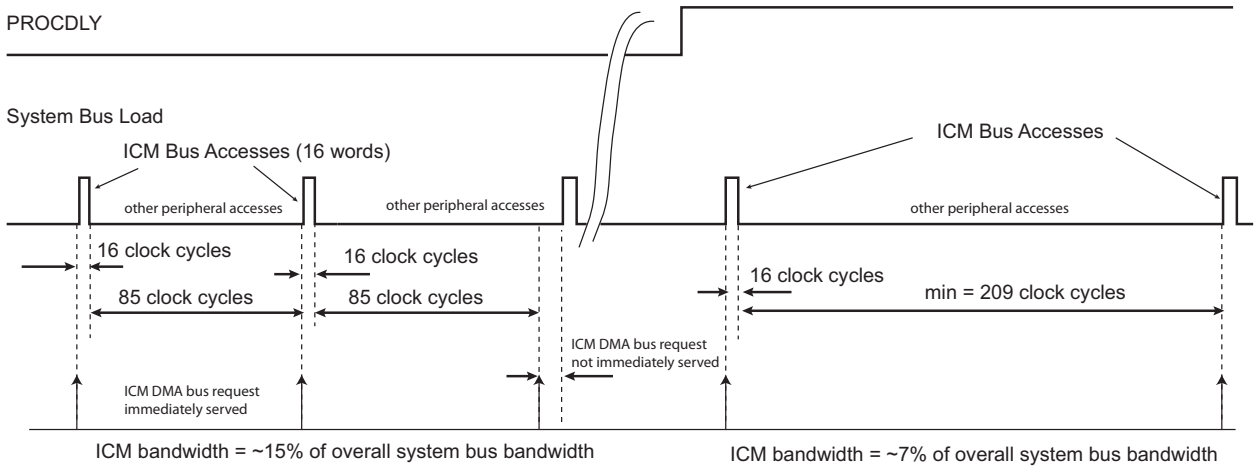
#### 57.5.4.2 Processing Period

The ICM engine has a core (SHA) inherent processing period that may result, depending on the application, in a significant bandwidth usage at system bus level. In some applications, it may be important to keep as much bandwidth as possible for the other peripherals (e.g. CPU, DMA). The ICM SHA engine processing period can be configured to reduce the bandwidth required by writing ICM_RCFG.PROCDLY=1.

In SHA1 mode, the shortest processing period is 85 clock cycles + 2 clock cycles for start command synchronization (ICM_RCFG.PROCDLY=0). The longest period is 209 clock cycles + 2 clock cycles when ICM_RCFG.PROCDLY=1 (see the figure below).

In SHA256 or SHA224 mode, the shortest processing period is 72 clock cycles + 2 clock cycles for start command synchronization. The longest period is 194 clock cycles + 2 clock cycles.

**Figure 57-6. Bandwidth Usage in SHA1 Mode**




### 57.5.5 ICM Automatic Monitoring Mode

ICM_CFG.ASCD is used to activate the ICM Automatic Monitoring mode. When ICM_CFG.ASCD is set and bits CDWBN and EOM in ICM.RCFG equal 0, the ICM performs the following actions:

1. The ICM passes through the Main List once to calculate the message digest of the monitored area.
2. When WRAP = 1 in ICM_RCFG, the ICM begins monitoring. CDWBN in ICM_RCFG is now automatically set and EOM is cleared. These bits have no effect during the monitoring period that ends when EOM is set.

### 57.5.6 Programming the ICM

**Table 57-6. Region Attributes**

| Transfer Type | | Main List | ICM_RCFG | | | ICM_RNEXT | Comments |
|---|---|---|---|---|---|---|---|
| | | | CDWBN | WRAP | EOM | NEXT | |
| Single Region | Contiguous list of blocks Digest written to memory Monitoring disabled | 1 item | 0 | 0 | 1 | 0 | The Main List contains only one descriptor. The Secondary List is empty for that descriptor. The digest is computed and saved to memory. |
| | Non-contiguous list of blocks Digest written to memory Monitoring disabled | 1 item | 0 | 0 | 1 | Secondary List address of the current region identifier | The Main List contains only one descriptor. The Secondary List describes the layout of the non-contiguous region. |
| | Contiguous list of blocks Digest comparison enabled Monitoring enabled | 1 item | 1 | 1 | 0 | 0 | When the hash computation is terminated, the digest is compared with the one saved in memory. |

.....continued

| Transfer Type | | Main List | ICM_RCFG | | | ICM_RNEXT | Comments |
|---|---|---|---|---|---|---|---|
| | | | CDWBN | WRAP | EOM | NEXT | |
| Multiple Regions | Contiguous list of blocks Digest written to memory Monitoring disabled | More than one item | 0 | 0 | 1 for the last, 0 otherwise | 0 | ICM passes through the list once. |
| | Contiguous list of blocks Digest comparison is enabled Monitoring is enabled | More than one item | 1 | 1 for the last, 0 otherwise | 0 | 0 | ICM performs active monitoring of the regions. If a mismatch occurs, an interrupt is raised. |
| | Non-contiguous list of blocks Digest is written to memory Monitoring is disabled | More than one item | 0 | 0 | 1 | Secondary List address | ICM performs hashing and saves digests to the Hash area. |
| | Non-contiguous list of blocks Digest comparison is enabled Monitoring is enabled | More than one item | 1 | 1 | 0 | Secondary List address | ICM performs data gathering on a per region basis. |

### 57.5.7 Security Features

When an undefined register access occurs, the URAD bit in the Interrupt Status Register (ICM_ISR) is set if unmasked. Its source is then reported in the Undefined Access Status Register (ICM_UASR). Only the first undefined register access is available through the ICM_UASR.URAT field.

Several kinds of unspecified register accesses can occur:

- Unspecified structure member set to one detected when the descriptor is loaded
- Configuration register (ICM_CFG) modified during active monitoring
- Descriptor register (ICM_DSCR) modified during active monitoring
- Hash register (ICM_HASH) modified during active monitoring
- Write-only register read access

The URAD bit and the URAT field can only be reset by writing a 1 to the ICM_CTRL.SWRST bit.

## 57.6 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | ICM_CFG | 7:0 | BBC[3:0] | | | | | SLBDIS | EOMDIS | WBDIS |
| | | 15:8 | UALGO[2:0] | | | UIHASH | | | DUALBUFF | ASCD |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | ICM_CTRL | 7:0 | REHASH[3:0] | | | | | SWRST | DISABLE | ENABLE |
| | | 15:8 | RMEN[3:0] | | | | RMDIS[3:0] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | ICM_SR | 7:0 | | | | | | | | ENABLE |
| | | 15:8 | RMDIS[3:0] | | | | RAWRMDIS[3:0] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | ICM_IER | 7:0 | RDM[3:0] | | | | RHC[3:0] | | | |
| | | 15:8 | RWC[3:0] | | | | RBE[3:0] | | | |
| | | 23:16 | RSU[3:0] | | | | REC[3:0] | | | |
| | | 31:24 | | | | | | | | URAD |
| 0x14 | ICM_IDR | 7:0 | RDM[3:0] | | | | RHC[3:0] | | | |
| | | 15:8 | RWC[3:0] | | | | RBE[3:0] | | | |
| | | 23:16 | RSU[3:0] | | | | REC[3:0] | | | |
| | | 31:24 | | | | | | | | URAD |
| 0x18 | ICM_IMR | 7:0 | RDM[3:0] | | | | RHC[3:0] | | | |
| | | 15:8 | RWC[3:0] | | | | RBE[3:0] | | | |
| | | 23:16 | RSU[3:0] | | | | REC[3:0] | | | |
| | | 31:24 | | | | | | | | URAD |
| 0x1C | ICM_ISR | 7:0 | RDM[3:0] | | | | RHC[3:0] | | | |
| | | 15:8 | RWC[3:0] | | | | RBE[3:0] | | | |
| | | 23:16 | RSU[3:0] | | | | REC[3:0] | | | |
| | | 31:24 | | | | | | | | URAD |
| 0x20 | ICM_UASR | 7:0 | | | | | | URAT[2:0] | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | ICM_DSCR | 7:0 | DASA[1:0] | | | | | | | |
| | | 15:8 | DASA[9:2] | | | | | | | |
| | | 23:16 | DASA[17:10] | | | | | | | |
| | | 31:24 | DASA[25:18] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x34 | ICM_HASH | 7:0 | HASA[0:0] | | | | | | | |
| | | 15:8 | HASA[8:1] | | | | | | | |
| | | 23:16 | HASA[16:9] | | | | | | | |
| | | 31:24 | HASA[24:17] | | | | | | | |
| 0x38 | ICM_UIHVAL0 | 7:0 | VAL[7:0] | | | | | | | |
| | | 15:8 | VAL[15:8] | | | | | | | |
| | | 23:16 | VAL[23:16] | | | | | | | |
| | | 31:24 | VAL[31:24] | | | | | | | |
| 0x3C | ICM_UIHVAL1 | 7:0 | VAL[7:0] | | | | | | | |
| | | 15:8 | VAL[15:8] | | | | | | | |
| | | 23:16 | VAL[23:16] | | | | | | | |
| | | 31:24 | VAL[31:24] | | | | | | | |
| 0x40 | ICM_UIHVAL2 | 7:0 | VAL[7:0] | | | | | | | |
| | | 15:8 | VAL[15:8] | | | | | | | |
| | | 23:16 | VAL[23:16] | | | | | | | |
| | | 31:24 | VAL[31:24] | | | | | | | |
| 0x44 | ICM_UIHVAL3 | 7:0 | VAL[7:0] | | | | | | | |
| | | 15:8 | VAL[15:8] | | | | | | | |
| | | 23:16 | VAL[23:16] | | | | | | | |
| | | 31:24 | VAL[31:24] | | | | | | | |
| 0x48 | ICM_UIHVAL4 | 7:0 | VAL[7:0] | | | | | | | |
| | | 15:8 | VAL[15:8] | | | | | | | |
| | | 23:16 | VAL[23:16] | | | | | | | |
| | | 31:24 | VAL[31:24] | | | | | | | |
| 0x4C | ICM_UIHVAL5 | 7:0 | VAL[7:0] | | | | | | | |
| | | 15:8 | VAL[15:8] | | | | | | | |
| | | 23:16 | VAL[23:16] | | | | | | | |
| | | 31:24 | VAL[31:24] | | | | | | | |
| 0x50 | ICM_UIHVAL6 | 7:0 | VAL[7:0] | | | | | | | |
| | | 15:8 | VAL[15:8] | | | | | | | |
| | | 23:16 | VAL[23:16] | | | | | | | |
| | | 31:24 | VAL[31:24] | | | | | | | |
| 0x54 | ICM_UIHVAL7 | 7:0 | VAL[7:0] | | | | | | | |
| | | 15:8 | VAL[15:8] | | | | | | | |
| | | 23:16 | VAL[23:16] | | | | | | | |
| | | 31:24 | VAL[31:24] | | | | | | | |

### 57.6.1 ICM Configuration Register

**Name:** ICM_CFG
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UALGO[2:0] | | | UIHASH | | | DUALBUFF | ASCD |
| Access | R/W | R/W | R/W | R/W | | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BBC[3:0] | | | | | SLBDIS | EOMDIS | WBDIS |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bits 15:13 – UALGO[2:0]** User SHA Algorithm

| Value | Name | Description |
|---|---|---|
| 0 | SHA1 | SHA1 algorithm processed |
| 1 | SHA256 | SHA256 algorithm processed |
| 4 | SHA224 | SHA224 algorithm processed |

**Bit 12 – UIHASH** User Initial Hash Value

| Value | Description |
|---|---|
| 0 | The secure hash standard provides the initial hash value. |
| 1 | The initial hash value is programmable. Field UALGO provides the SHA algorithm. The ALGO field of the ICM_RCFG structure member has no effect. |

**Bit 9 – DUALBUFF** Dual Input Buffer

| Value | Description |
|---|---|
| 0 | Dual Input Buffer mode is disabled. |
| 1 | Dual Input Buffer mode is enabled (better performances, higher bandwidth required on system bus). |

**Bit 8 – ASCD** Automatic Switch To Compare Digest

| Value | Description |
|---|---|
| 0 | Automatic monitoring mode is disabled. |
| 1 | The ICM passes through the Main List once to calculate the message digest of the monitored area. When WRAP = 1 in ICM_RCFG, the ICM begins monitoring. |

**Bits 7:4 – BBC[3:0]** Bus Burden Control
This field is used to control the burden of the ICM system bus. The number of system clock cycles between the end of the current processing and the next block transfer is set to $2^{BBC}$. Up to 32,768 cycles can be inserted.

**Bit 2 – SLBDIS** Secondary List Branching Disable

| Value | Description |
|---|---|
| 0 | Branching to the Secondary List is permitted. |
| 1 | Branching to the Secondary List is forbidden. The NEXT field of the ICM_RNEXT structure member has no effect and is always considered as zero. |

**Bit 1 – EOMDIS** End of Monitoring Disable

| Value | Description |
|---|---|
| 0 | End of Monitoring is permitted. |
| 1 | End of Monitoring is forbidden. The EOM bit of the ICM_RCFG structure member has no effect. |

**Bit 0 – WBDIS** Write Back Disable
When ASCD is set, WBDIS has no effect.

| Value | Description |
|---|---|
| 0 | Write Back operations are permitted. |
| 1 | Write Back operations are forbidden. Context register CDWBN bit is internally set to one and cannot be modified by a linked list element. ICM_RCFG.CDWBN has no effect. |

### 57.6.2 ICM Control Register

**Name:** ICM_CTRL
**Offset:** 0x04
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RMEN[3:0] | | | | RMDIS[3:0] | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | – | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | REHASH[3:0] | | | | | SWRST | DISABLE | ENABLE |
| Access | W | W | W | W | | W | W | W |
| Reset | 0 | 0 | 0 | – | | – | – | – |

**Bits 15:12 – RMEN[3:0]** Region Monitoring Enable
Monitoring is activated by default.

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | When bit RMEN[i] is set to one, the monitoring of region with identifier i is activated. |

**Bits 11:8 – RMDIS[3:0]** Region Monitoring Disable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | When bit RMDIS[i] is set to one, the monitoring of region with identifier i is disabled. |

**Bits 7:4 – REHASH[3:0]** Recompute Internal Hash

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | When REHASH[i] is set to one, Region i digest is re-computed. This bit is only available when region monitoring is disabled. |

**Bit 2 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Resets the ICM. |

**Bit 1 – DISABLE** ICM Disable Register

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | The ICM is disabled. If a region is active, this region is terminated. |

**Bit 0 – ENABLE** ICM Enable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | When set to one, the ICM is activated. |

### 57.6.3 ICM Status Register

**Name:** ICM_SR
**Offset:** 0x08
**Reset:** –
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RMDIS[3:0] | | | | RAWRMDIS[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | – | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ENABLE |
| Access | | | | | | | | R |
| Reset | | | | | | | | – |

**Bits 15:12 – RMDIS[3:0]** Region Monitoring Disabled Status

| Value | Description |
|---|---|
| 0 | Region i is being monitored (occurs after integrity check value has been calculated and written to Hash area). |
| 1 | Region i monitoring is not being monitored. |

**Bits 11:8 – RAWRMDIS[3:0]** Region Monitoring Disabled Raw Status

| Value | Description |
|---|---|
| 0 | Region i monitoring has been activated by writing a 1 in RMEN[i] of ICM_CTRL. |
| 1 | Region i monitoring has been deactivated by writing a 1 in RMDIS[i] of ICM_CTRL. |

**Bit 0 – ENABLE** ICM Enable Register

| Value | Description |
|---|---|
| 0 | ICM is disabled. |
| 1 | ICM is activated. |

### 57.6.4 ICM Interrupt Enable Register

**Name:** ICM_IER
**Offset:** 0x10
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RSU[3:0] | | | | REC[3:0] | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | – | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RWC[3:0] | | | | RBE[3:0] | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | – | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RDM[3:0] | | | | RHC[3:0] | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | – | 0 | 0 | 0 | – |

**Bit 24 – URAD** Undefined Register Access Detection Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The Undefined Register Access interrupt is enabled. |

**Bits 23:20 – RSU[3:0]** Region Status Updated Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RSU[i] is set to one, the region i Status Updated interrupt is enabled. |

**Bits 19:16 – REC[3:0]** Region End bit Condition Detected Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When REC[i] is set to one, the region i End bit Condition interrupt is enabled. |

**Bits 15:12 – RWC[3:0]** Region Wrap Condition detected Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RWC[i] is set to one, the Region i Wrap Condition interrupt is enabled. |

**Bits 11:8 – RBE[3:0]** Region Bus Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RBE[i] is set to one, the Region i Bus Error interrupt is enabled. |

**Bits 7:4 – RDM[3:0]** Region Digest Mismatch Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RDM[i] is set to one, the Region i Digest Mismatch interrupt is enabled. |

**Bits 3:0 – RHC[3:0]** Region Hash Completed Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RHC[i] is set to one, the Region i Hash Completed interrupt is enabled. |

### 57.6.5 ICM Interrupt Disable Register

**Name:** ICM_IDR
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RSU[3:0] | | | | REC[3:0] | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | – | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RWC[3:0] | | | | RBE[3:0] | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | – | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RDM[3:0] | | | | RHC[3:0] | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | – | 0 | 0 | 0 | – |

**Bit 24 – URAD** Undefined Register Access Detection Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Undefined Register Access Detection interrupt is disabled. |

**Bits 23:20 – RSU[3:0]** Region Status Updated Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RSU[i] is set to one, the region i Status Updated interrupt is disabled. |

**Bits 19:16 – REC[3:0]** Region End bit Condition detected Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When REC[i] is set to one, the region i End bit Condition interrupt is disabled. |

**Bits 15:12 – RWC[3:0]** Region Wrap Condition Detected Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RWC[i] is set to one, the Region i Wrap Condition interrupt is disabled. |

**Bits 11:8 – RBE[3:0]** Region Bus Error Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RBE[i] is set to one, the Region i Bus Error interrupt is disabled. |

**Bits 7:4 – RDM[3:0]** Region Digest Mismatch Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RDM[i] is set to one, the Region i Digest Mismatch interrupt is disabled. |

**Bits 3:0 – RHC[3:0]** Region Hash Completed Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RHC[i] is set to one, the Region i Hash Completed interrupt is disabled. |

### 57.6.6 ICM Interrupt Mask Register

**Name:** ICM_IMR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RSU[3:0] | | | | REC[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RWC[3:0] | | | | RBE[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RDM[3:0] | | | | RHC[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 24 – URAD** Undefined Register Access Detection Interrupt Mask

| Value | Description |
|---|---|
| 0 | Interrupt is disabled |
| 1 | Interrupt is enabled. |

**Bits 23:20 – RSU[3:0]** Region Status Updated Interrupt Mask

| Value | Description |
|---|---|
| 0 | When RSU[i] is set to zero, the interrupt is disabled for region i. |
| 1 | When RSU[i] is set to one, the interrupt is enabled for region i. |

**Bits 19:16 – REC[3:0]** Region End Bit Condition Detected Interrupt Mask

| Value | Description |
|---|---|
| 0 | When REC[i] is set to zero, the interrupt is disabled for region i. |
| 1 | When REC[i] is set to one, the interrupt is enabled for region i. |

**Bits 15:12 – RWC[3:0]** Region Wrap Condition Detected Interrupt Mask

| Value | Description |
|---|---|
| 0 | When RWC[i] is set to zero, the interrupt is disabled for region i. |
| 1 | When RWC[i] is set to one, the interrupt is enabled for region i. |

**Bits 11:8 – RBE[3:0]** Region Bus Error Interrupt Mask

| Value | Description |
|---|---|
| 0 | When RBE[i] is set to zero, the interrupt is disabled for region i. |
| 1 | When RBE[i] is set to one, the interrupt is enabled for region i. |

**Bits 7:4 – RDM[3:0]** Region Digest Mismatch Interrupt Mask

| Value | Description |
|---|---|
| 0 | When RDM[i] is set to zero, the interrupt is disabled for region i. |
| 1 | When RDM[i] is set to one, the interrupt is enabled for region i. |

**Bits 3:0 – RHC[3:0]** Region Hash Completed Interrupt Mask

| Value | Description |
|---|---|
| 0 | When RHC[i] is set to zero, the interrupt is disabled for region i. |
| 1 | When RHC[i] is set to one, the interrupt is enabled for region i. |

### 57.6.7 ICM Interrupt Status Register

**Name:** ICM_ISR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RSU[3:0] | | | | REC[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RWC[3:0] | | | | RBE[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RDM[3:0] | | | | RHC[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 24 – URAD** Undefined Register Access Detection Status
The URAD bit is only reset by the SWRST bit in ICM_CTRL.

The URAT field in ICM_UASR indicates the unspecified access type.

| Value | Description |
|---|---|
| 0 | No undefined register access has been detected since the last SWRST. |
| 1 | At least one undefined register access has been detected since the last SWRST. |

**Bits 23:20 – RSU[3:0]** Region Status Updated Detected
When RSU[i] is set, it indicates that a region status updated condition has been detected.

**Bits 19:16 – REC[3:0]** Region End Bit Condition Detected
When REC[i] is set, it indicates that an end bit condition has been detected.

**Bits 15:12 – RWC[3:0]** Region Wrap Condition Detected
When RWC[i] is set, it indicates that a wrap condition has been detected.

**Bits 11:8 – RBE[3:0]** Region Bus Error
When RBE[i] is set, it indicates that a bus error has been detected while hashing memory region i.

**Bits 7:4 – RDM[3:0]** Region Digest Mismatch
When RDM[i] is set, it indicates that there is a digest comparison mismatch between the hash value of the region with identifier i and the reference value located in the Hash Area.

**Bits 3:0 – RHC[3:0]** Region Hash Completed
When RHC[i] is set, it indicates that the ICM has completed the region with identifier i.

### 57.6.8 ICM Undefined Access Status Register

**Name:** ICM_UASR
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | URAT[2:0] | | |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 2:0 – URAT[2:0]** Undefined Register Access Trace
Only the first Undefined Register Access Trace is available through the URAT field.

The URAT field is only reset by the SWRST bit in the ICM_CTRL register.

| Value | Name | Description |
|---|---|---|
| 0 | UNSPEC_STRUCT_MEMBER | Unspecified structure member set to one detected when the descriptor is loaded. |
| 1 | ICM_CFG_MODIFIED | ICM_CFG modified during active monitoring. |
| 2 | ICM_DSCR_MODIFIED | ICM_DSCR modified during active monitoring. |
| 3 | ICM_HASH_MODIFIED | ICM_HASH modified during active monitoring |
| 4 | READ_ACCESS | Write-only register read access |

### 57.6.9 ICM Descriptor Area Start Address Register

**Name:** ICM_DSCR
**Offset:** 0x30
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DASA[25:18] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DASA[17:10] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DASA[9:2] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DASA[1:0] | | | | | | | |
| Access | R/W | R/W | | | | | | |
| Reset | 0 | 0 | | | | | | |

**Bits 31:6 – DASA[25:0]** Descriptor Area Start Address
The start address is a multiple of the total size of the data structure (64 bytes).

### 57.6.10 ICM Hash Area Start Address Register

**Name:** ICM_HASH
**Offset:** 0x34
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | HASA[24:17] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | HASA[16:9] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | HASA[8:1] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | HASA[0:0] | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0 | | | | | | | |

**Bits 31:7 – HASA[24:0]** Hash Area Start Address
This field points at the Hash memory location. The address must be a multiple of 128 bytes.

### 57.6.11 ICM User Initial Hash Value Register

**Name:** ICM_UIHVALx
**Offset:** 0x38 + x*0x04 [x=0..7]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | VAL[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | VAL[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | VAL[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | VAL[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – VAL[31:0]** Initial Hash Value
When ICM_CFG.UIHASH is set, the Initial Hash Value is user-programmable.

To meet the desired standard, use the following example values.

For ICM_UIHVAL0 field:

| Example | Comment |
|---|---|
| 0x67452301 | SHA1 algorithm |
| 0xC1059ED8 | SHA224 algorithm |
| 0x6A09E667 | SHA256 algorithm |

For ICM_UIHVAL1 field:

| Example | Comment |
|---|---|
| 0xEFCDAB89 | SHA1 algorithm |
| 0x367CD507 | SHA224 algorithm |
| 0xBB67AE85 | SHA256 algorithm |

For ICM_UIHVAL2 field:

| Example | Comment |
|---|---|
| 0x98BADCFE | SHA1 algorithm |
| 0x3070DD17 | SHA224 algorithm |
| 0x3C6EF372 | SHA256 algorithm |

For ICM_UIHVAL3 field:

| Example | Comment |
|---|---|
| 0x10325476 | SHA1 algorithm |
| 0xF70E5939 | SHA224 algorithm |
| 0xA54FF53A | SHA256 algorithm |

For ICM_UIHVAL4 field:

| Example | Comment |
|---|---|
| 0xC3D2E1F0 | SHA1 algorithm |
| 0xFFC00B31 | SHA224 algorithm |
| 0x510E527F | SHA256 algorithm |

For ICM_UIHVAL5 field:

| Example | Comment |
|---|---|
| 0x68581511 | SHA224 algorithm |
| 0x9B05688C | SHA256 algorithm |

For ICM_UIHVAL6 field:

| Example | Comment |
|---|---|
| 0x64F98FA7 | SHA224 algorithm |
| 0x1F83D9AB | SHA256 algorithm |

For ICM_UIHVAL7 field:

| Example | Comment |
|---|---|
| 0xBEFA4FA4 | SHA224 algorithm |
| 0x5BE0CD19 | SHA256 algorithm |

Example of Initial Value for SHA-1 Algorithm

| Register Address | Address Offset / Byte Lane | | | |
|---|---|---|---|---|
| | 0x3 / 31:24 | 0x2 / 23:16 | 0x1 / 15:8 | 0x0 / 7:0 |
| 0x000 ICM_UIHVAL0 | 01 | 23 | 45 | 67 |

.....continued

| Register Address | Address Offset / Byte Lane | | | |
|---|---|---|---|---|
| | 0x3 / 31:24 | 0x2 / 23:16 | 0x1 / 15:8 | 0x0 / 7:0 |
| 0x004 ICM_UIHVAL1 | 89 | ab | cd | ef |
| 0x008 ICM_UIHVAL2 | fe | dc | ba | 98 |
| 0x00C ICM_UIHVAL3 | 76 | 54 | 32 | 10 |
| 0x010 ICM_UIHVAL4 | f0 | e1 | d2 | c3 |

# 58. Advanced Encryption Standard Bridge (AESB)

## 58.1 Description

The Advanced Encryption Standard Bridge (AESB) is intended to provide on-the-fly off-chip memory encryption/decryption compliant with the American *FIPS (Federal Information Processing Standard) Publication 197* specification.

The AESB supports three confidentiality modes of operation for symmetrical key block cipher algorithms (ECB, CBC and CTR), as specified in the *NIST Special Publication 800-38A Recommendation*.

The AESB key is loaded by the software.

The 128-bit AESB key is stored in the AESB Key register made of four 32-bit write-only AESB Key Word registers (AESB_KEYWR0–3).

The 128-bit input data and initialization vector (for some modes) are each stored in four 32-bit registers (AESB_IDATARx and AESB_IVRx) which are all write-only.

As soon as the initialization vector, the input data and the key are configured, the encryption/decryption process may be started. Then the encrypted/decrypted data will be ready to be read out on the four 32-bit output data registers (AESB_ODATARx).

## 58.2 Embedded Characteristics

- On-The-Fly Off-Chip Memory Encryption/Decryption
- Compliant with FIPS Publication 197, Advanced Encryption Standard (AES)
- 128-bit Cryptographic Key
- On-The-Fly Encryption/Decryption
- 10 Clock Cycles Encryption/Decryption Inherent Processing Time
- Double Input Buffer Optimizes Runtime
- Support of the Three Standard Modes of Operation Specified in *NIST Special Publication 800-38A, Recommendation for Block Cipher Modes of Operation - Methods and Techniques*:
  - Electronic Code Book (ECB)
  - Cipher Block Chaining (CBC) including CBC-MAC
  - Counter (CTR)
- Last Output Data Mode Allows Optimized Message Authentication Code (MAC) Generation

## 58.3 Product Dependencies

### 58.3.1 Power Management

The AESB may be clocked through the Power Management Controller (PMC), so the programmer must first configure the PMC to enable the AESB clock.

### 58.3.2 Interrupt

The AESB interface has an interrupt line connected to the Interrupt Controller.

Handling the AESB interrupt requires programming the Interrupt Controller before configuring the AESB.

## 58.4 Functional Description

The Advanced Encryption Standard Bridge (AESB) specifies a FIPS-approved cryptographic algorithm that can be used to protect electronic data. The AESB algorithm is a symmetric block cipher that can encrypt (encipher) and decrypt (decipher) information.

Encryption converts data to an unintelligible form called ciphertext. Decrypting the ciphertext converts the data back into its original form, called plaintext. CIPHER in the AESB Mode register (AESB_MR) allows selection between the encryption and the decryption processes.

The AESB is capable of using cryptographic keys of 128 bits to encrypt and decrypt data in blocks of 128 bits. This 128-bit key is defined in the Key registers (AESB_KEYWRx).

The input to the encryption processes of the CBC mode includes, in addition to the plaintext, a 128-bit data block called the initialization vector (IV), which must be set in the Initialization Vector Registers (AESB_IVRx). The initialization vector is used in an initial step in the encryption of a message and in the corresponding decryption of the message. The Initialization Vector registers are also used by the CTR mode to set the counter value.

### 58.4.1 Operating Modes

The AESB supports the following modes of operation:

- ECB—Electronic Code Book
- CBC—Cipher Block Chaining
- CTR—Counter

The data pre-processing, post-processing and data chaining for the operating modes are performed automatically. Refer to *NIST Special Publication 800-38A Recommendation* for more complete information.

The modes are selected in AESB_MR.OPMOD.

In CTR mode, the size of the block counter embedded in the module is 16 bits. Therefore, there is a rollover after processing 1 megabyte of data. If the file to be processed is greater than 1 megabyte, this file must be split into fragments of 1 megabyte or less for the first fragment if the initial value of the counter is greater than 0. Prior to loading the first fragment into AESB_IDATARx registers, the AESB_IVRx registers must be cleared. For any fragment, after the transfer is completed and prior to transferring the next fragment, AESB_IVR0 must be programmed so that the fragment number (0 for the first fragment, 1 for the second one, and so on) is written in the 16 MSB of AESB_IVR0.

If the initial value of the counter is greater than 0 and the data buffer size to be processed is greater than 1 megabyte, the size of the first fragment to be processed must be 1 megabyte minus 16x(initial value) to prevent a rollover of the internal 1-bit counter.

### 58.4.2 Double Input Buffer

The input data register can be double-buffered to reduce the runtime of large files.

This mode allows writing a new message block when the previous message block is being processed.

AESB_MR.DUALBUFF must be set to '1' to access the double buffer.

### 58.4.3 Start Modes

AESB_MR.SMOD allows selection of the Encryption or Decryption Start mode.

#### 58.4.3.1 Manual Mode

The sequence is as follows:

1. Write AESB_MR with all required fields, including but not limited to SMOD and OPMOD.
2. Write the 128-bit key in the Key registers (AESB_KEYWRx).
3. Write the initialization vector (or counter) in the Initialization Vector registers (AESB_IVRx).
   **Note:** The Initialization Vector registers concern all modes except ECB.
4. Set DATRDY (Data Ready) in the AESB Interrupt Enable register (AESB_IER) depending on whether an interrupt is required, or not, at the end of processing.
5. Write the data to be encrypted/decrypted in the authorized Input Data registers (see the table below).

**Table 58-1. Authorized Input Data Registers**

| Operating Mode | Input Data Registers to Write |
|---|---|
| ECB | All |
| CBC | All |
| CTR | All |

6. Set the START bit in the AESB Control register (AESB_CR) to begin the encryption or decryption process.
7. When processing is complete, DATRDY in the AESB Interrupt Status register (AESB_ISR) rises. If an interrupt has been enabled by setting AESB_IER.DATRDY, the interrupt line of the AESB is activated.
8. When the software reads one of the Output Data registers (AESB_ODATARx), AESB_ISR.DATRDY is automatically cleared.

#### 58.4.3.2 Auto Mode

In Auto mode, as soon as the correct number of Input Data registers is written, processing starts automatically without any action in AESB_CR.

### 58.4.4 Last Output Data Mode

Last Output Data mode is used to generate cryptographic checksums on data (MAC) by means of a cipher block chaining encryption algorithm (e.g., the CBC-MAC algorithm).

After each end of encryption/decryption, the output data are available on the output data registers for Manual and Auto modes.

AESB_MR.LOD allows retrieval of only the last data of several encryption/decryption processes.

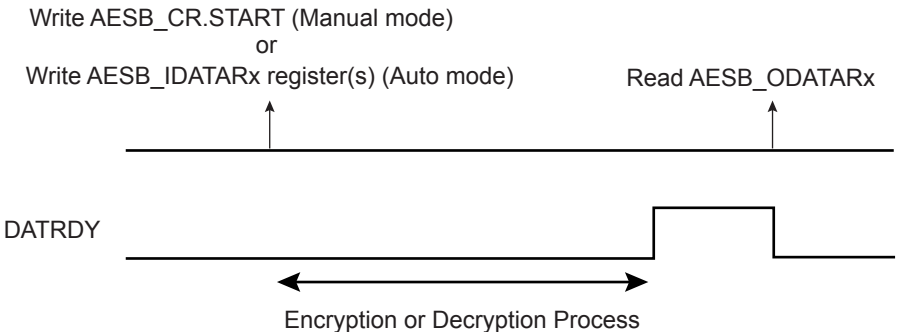
Those data are only available on the Output Data registers (AESB_ODATARx).

### 58.4.5 Manual and Auto Modes

#### 58.4.5.1 If AESB_MR.LOD = 0

AESB_ISR.DATRDY is cleared when at least one of the Output Data registers is read (see the figure below).

**Figure 58-1. Manual and Auto Modes with AESB_MR.LOD = 0**

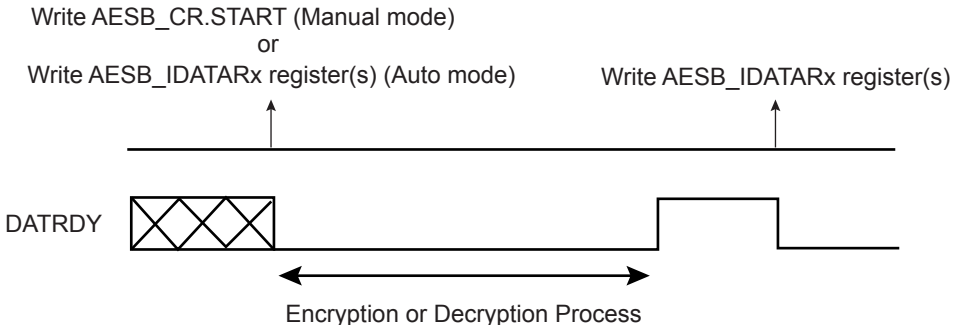



If the user does not want to read the output data registers between each encryption/decryption, AESB_ISR.DATRDY will not be cleared. If AESB_ISR.DATRDY is not cleared, the user cannot know the end of the following encryptions/decryptions.

#### 58.4.5.2 If AESB_MR.LOD = 1

AESB_ISR.DATRDY is cleared when at least one Input Data register is written, so before the start of a new transfer (see the figure below). No more Output Data register reads are necessary between consecutive encryptions/decryptions.

**Figure 58-2. Manual and Auto Modes with AESB_MR.LOD = 1**




### 58.4.6 Automatic Bridge Mode

#### 58.4.6.1 Description

The Automatic Bridge mode, when the AESB block is connected between the system bus and a DDR port and the QSPI, provides automatic encryption/decryption without any action on the part of the user. For Automatic Bridge mode, AESB_MR.OPMOD must be configured to 0x4 (see AESB Mode Register). If AESB_MR.AAHB is set and AESB_MR.OPMOD = 0x4, there is no compliance with the standard CTR mode of operation.

In case of write transfer, this mode automatically encrypts the data before writing it to the final slave destination. In case of read transfer, this mode automatically decrypts the data read from the target slave before putting it on the system bus.

Therefore, this mode does not work if the automatically encrypted data is moved at another address outside of the AESB. This means that for a given data, the encrypted value is not the same if written at different addresses.

#### 58.4.6.2 Configuration

The Automatic Bridge mode can be enabled by setting AESB_MR.AAHB.

The IV (Initialization Vector) field of the AESB Initialization Vector register x (AESB_IVRx) can be used to add a nonce in the encryption process in order to bring even more security (ignored if not filled). In this

case, any value encrypted with a given nonce can only be decrypted with this nonce. If another nonce is set for the AESB_IVRx.IV, any value encrypted with the previous nonce can no longer be decrypted (see AESB Initialization Vector Register x).

Dual buffer usage (AESB_MR.DUALBUFF='1') is recommended for improved performance.

### 58.4.7 Security Features

#### 58.4.7.1 Unspecified Register Access Detection

When an unspecified register access occurs, AESB_ISR.URAD rises. Its source is then reported in AESB_ISR.URAT. Only the last unspecified register access is available through the URAT field.

Several kinds of unspecified register accesses can occur:

- Input Data register written during the data processing when SMOD = IDATAR0_START
- Output Data register read during data processing
- Mode register written during data processing
- Output Data register read during sub-keys generation
- Mode register written during sub-keys generation
- Write-only register read access

URAD and URAT can only be reset by AESB_CR.SWRST.

## 58.5 Register Summary

This is the start of your topic.

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | AESB_CR | 7:0 | | | | | | | | START |
| | | 15:8 | | | | | | | | SWRST |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x00 | AESB_IDATAR0 | 7:0 | IDATA[7:0] | | | | | | | |
| | | 15:8 | IDATA[15:8] | | | | | | | |
| | | 23:16 | IDATA[23:16] | | | | | | | |
| | | 31:24 | IDATA[31:24] | | | | | | | |
| 0x00 | AESB_ODATAR0 | 7:0 | ODATA[7:0] | | | | | | | |
| | | 15:8 | ODATA[15:8] | | | | | | | |
| | | 23:16 | ODATA[23:16] | | | | | | | |
| | | 31:24 | ODATA[31:24] | | | | | | | |
| 0x00 | AESB_IVR0 | 7:0 | IV[7:0] | | | | | | | |
| | | 15:8 | IV[15:8] | | | | | | | |
| | | 23:16 | IV[23:16] | | | | | | | |
| | | 31:24 | IV[31:24] | | | | | | | |
| 0x04 | AESB_MR | 7:0 | PROCDLY[3:0] | | | | DUALBUFF | AAHB | | CIPHER |
| | | 15:8 | LOD | OPMOD[2:0] | | | | | SMOD[1:0] | |
| | | 23:16 | CKEY[3:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | AESB_IDATAR1 | 7:0 | IDATA[7:0] | | | | | | | |
| | | 15:8 | IDATA[15:8] | | | | | | | |
| | | 23:16 | IDATA[23:16] | | | | | | | |
| | | 31:24 | IDATA[31:24] | | | | | | | |
| 0x04 | AESB_ODATAR1 | 7:0 | ODATA[7:0] | | | | | | | |
| | | 15:8 | ODATA[15:8] | | | | | | | |
| | | 23:16 | ODATA[23:16] | | | | | | | |
| | | 31:24 | ODATA[31:24] | | | | | | | |
| 0x04 | AESB_IVR1 | 7:0 | IV[7:0] | | | | | | | |
| | | 15:8 | IV[15:8] | | | | | | | |
| | | 23:16 | IV[23:16] | | | | | | | |
| | | 31:24 | IV[31:24] | | | | | | | |
| 0x08 | AESB_IDATAR2 | 7:0 | IDATA[7:0] | | | | | | | |
| | | 15:8 | IDATA[15:8] | | | | | | | |
| | | 23:16 | IDATA[23:16] | | | | | | | |
| | | 31:24 | IDATA[31:24] | | | | | | | |
| 0x08 | AESB_ODATAR2 | 7:0 | ODATA[7:0] | | | | | | | |
| | | 15:8 | ODATA[15:8] | | | | | | | |
| | | 23:16 | ODATA[23:16] | | | | | | | |
| | | 31:24 | ODATA[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x08 | AESB_IVR2 | 7:0 | IV[7:0] | | | | | | | |
| | | 15:8 | IV[15:8] | | | | | | | |
| | | 23:16 | IV[23:16] | | | | | | | |
| | | 31:24 | IV[31:24] | | | | | | | |
| 0x0C | AESB_IDATAR3 | 7:0 | IDATA[7:0] | | | | | | | |
| | | 15:8 | IDATA[15:8] | | | | | | | |
| | | 23:16 | IDATA[23:16] | | | | | | | |
| | | 31:24 | IDATA[31:24] | | | | | | | |
| 0x0C | AESB_ODATAR3 | 7:0 | ODATA[7:0] | | | | | | | |
| | | 15:8 | ODATA[15:8] | | | | | | | |
| | | 23:16 | ODATA[23:16] | | | | | | | |
| | | 31:24 | ODATA[31:24] | | | | | | | |
| 0x0C | AESB_IVR3 | 7:0 | IV[7:0] | | | | | | | |
| | | 15:8 | IV[15:8] | | | | | | | |
| | | 23:16 | IV[23:16] | | | | | | | |
| | | 31:24 | IV[31:24] | | | | | | | |
| 0x10 | AESB_IER | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | URAD |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | AESB_IDR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | URAD |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | AESB_IMR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | URAD |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | AESB_ISR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | URAT[3:0] | | | | | | | URAD |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | AESB_KEYWR0 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |
| 0x24 | AESB_KEYWR1 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |
| 0x28 | AESB_KEYWR2 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x2C | AESB_KEYWR3 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |

### 58.5.1 AESB Control Register

**Name:** AESB_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SWRST |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | START |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 8 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Resets the AESB. A software triggered hardware reset of the AESB interface is performed. |

**Bit 0 – START** Start Processing

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Starts manual encryption/decryption process |

### 58.5.2 AESB Mode Register

**Name:** AESB_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CKEY[3:0] | | | | | | | |
| Access | W | W | W | W | | | | |
| Reset | 0 | 0 | 0 | – | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LOD | OPMOD[2:0] | | | | | SMOD[1:0] | |
| Access | R/W | R/W | R/W | R/W | | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PROCDLY[3:0] | | | | DUALBUFF | AAHB | | CIPHER |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | 0 |

**Bits 23:20 – CKEY[3:0]** Key

| Value | Name | Description |
|---|---|---|
| 0xE | PASSWD | Must be written with 0xE the first time that AESB_MR is programmed. For subsequent programming of the AESB_MR, any value can be written, including that of 0xE.<br>Always reads as 0. |

**Bit 15 – LOD** Last Output Data Mode

| Value | Description |
|---|---|
| 0 | No effect.<br>After each end of encryption/decryption, the output data will be available either on the output data registers (Manual and Auto modes).<br>In Manual and Auto modes, AESB_ISR.DATRDY is cleared when at least one of the Output Data registers is read. |
| 1 | AESB_ISR.DATRDY is cleared when at least one of the Input Data registers is written.<br>No additional Output Data register reads are necessary between consecutive encryptions/decryptions (see Last Output Data Mode). |

**Bits 14:12 – OPMOD[2:0]** Operating Mode
Values which are not listed in the table must be considered as “reserved”.

For CBC-MAC operating mode, configure OPMOD to 0x1 (CBC) and set LOD to 1.

If OPMOD is set to 4 and AAHB = 1, there is no compliance with the standard CTR mode of operation.

| Value | Name | Description |
|---|---|---|
| 0 | ECB | Electronic Code Book mode |
| 1 | CBC | Cipher Block Chaining mode |
| 2 | – | Reserved |
| 3 | – | Reserved |
| 4 | CTR | Counter mode (16-bit internal counter) |

**Bits 9:8 – SMOD[1:0]** Start Mode

| Value | Name | Description |
|---|---|---|
| 0 | MANUAL_START | Manual mode |
| 1 | AUTO_START | Auto mode |
| 2 | IDATAR0_START | AESB_IDATAR0 access only Auto mode |

**Bits 7:4 – PROCDLY[3:0]** Processing Delay
Processing Time = 12 × (PROCDLY + 1)

The Processing Time represents the number of clock cycles that the AESB needs in order to perform one encryption/decryption .

**Note:** The best performance is achieved with PROCDLY = 0.

**Bit 3 – DUALBUFF** Dual Input Buffer

| Value | Name | Description |
|---|---|---|
| 0 | INACTIVE | AESB_IDATARx cannot be written during processing of previous block. |
| 1 | ACTIVE | AESB_IDATARx can be written during processing of previous block when SMOD = 2. It speeds up the overall runtime of large files. |

**Bit 2 – AAHB** Automatic Bridge Mode

| Value | Description |
|---|---|
| 0 | Automatic Bridge mode disabled. |
| 1 | Automatic Bridge mode enabled. |

**Bit 0 – CIPHER** Processing Mode

| Value | Description |
|---|---|
| 0 | Decrypts data. |
| 1 | Encrypts data. |

### 58.5.3 AESB Interrupt Enable Register

**Name:** AESB_IER
**Offset:** 0x10
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 8 – URAD** Unspecified Register Access Detection Interrupt Enable

**Bit 0 – DATRDY** Data Ready Interrupt Enable

### 58.5.4 AESB Interrupt Disable Register

**Name:** AESB_IDR
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 8 – URAD** Unspecified Register Access Detection Interrupt Disable

**Bit 0 – DATRDY** Data Ready Interrupt Disable

### 58.5.5 AESB Interrupt Mask Register

**Name:** AESB_IMR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 8 – URAD** Unspecified Register Access Detection Interrupt Mask

**Bit 0 – DATRDY** Data Ready Interrupt Mask

### 58.5.6 AESB Interrupt Status Register

**Name:** AESB_ISR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | URAT[3:0] | | | | | | | URAD |
| Access | R | R | R | R | | | | R |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 15:12 – URAT[3:0]** Unspecified Register Access
Only the last Unspecified Register Access Type is available through URAT.

URAT field is reset only by AESB_CR.SWRST.

| Value | Name | Description |
|---|---|---|
| 0x0 | IDR_WR_PROCESSING | Input Data register written during the data processing when SMOD = 0x2 mode |
| 0x1 | ODR_RD_PROCESSING | Output Data register read during the data processing |
| 0x2 | MR_WR_PROCESSING | Mode register written during the data processing |
| 0x3 | ODR_RD_SUBKGEN | Output Data register read during the sub-keys generation |
| 0x4 | MR_WR_SUBKGEN | Mode register written during the sub-keys generation |
| 0x5 | WOR_RD_ACCESS | Write-only register read access |

**Bit 8 – URAD** Unspecified Register Access Detection Status
URAD is reset only by AESB_CR.SWRST.

| Value | Description |
|---|---|
| 0 | No unspecified register access has been detected since the last SWRST. |
| 1 | At least one unspecified register access has been detected since the last SWRST. |

**Bit 0 – DATRDY** Data Ready
DATRDY is cleared when a Manual encryption/decryption occurs (AESB_CR.START) or when a software triggered hardware reset of the AESB interface is performed (AESB_CR.SWRST).

AESB_MR.LOD = 0: In Manual and Auto modes, DATRDY can also be cleared when at least one of the Output Data registers is read.

AESB_MR.LOD = 1: In Manual and Auto modes, DATRDY can also be cleared when at least one of the Input Data registers is written.

| Value | Description |
|---|---|
| 0 | Output data not valid. |
| 1 | Encryption or decryption process is completed. |

### 58.5.7 AESB Key Word Register x

**Name:** AESB_KEYWRx
**Offset:** 0x20 + x*0x04 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEYW[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEYW[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | KEYW[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | KEYW[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – KEYW[31:0]** Key Word
The four 32-bit Key Word registers set the 128-bit cryptographic key used for encryption/decryption.

AESB_KEYWR0 corresponds to the first word of the key, AESB_KEYWR3 to the last one.

These registers are write-only to prevent the key from being read by another application.

### 58.5.8 AESB Input Data Register x

**Name:** AESB_IDATARx
**Offset:** 0x00 + x*0x04 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – IDATA[31:0]** Input Data Word
The four 32-bit Input Data registers set the 128-bit data block used for encryption/decryption.

AESB_IDATAR0 corresponds to the first word of the data to be encrypted/decrypted, AESB_IDATAR3 to the last one.

These registers are write-only to prevent the input data from being read by another application.

### 58.5.9 AESB Output Data Register x

**Name:** AESB_ODATARx
**Offset:** 0x00 + x*0x04 [x=0..3]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – ODATA[31:0]** Output Data
The four 32-bit Output Data registers contain the 128-bit data block that has been encrypted/decrypted.

AESB_ODATAR0 corresponds to the first word, AESB_ODATAR3 to the last one.

### 58.5.10 AESB Initialization Vector Register x

**Name:** AESB_IVRx
**Offset:** 0x00 + x*0x04 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IV[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IV[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IV[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IV[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – IV[31:0]** Initialization Vector
The four 32-bit Initialization Vector registers set the 128-bit Initialization Vector data block that is used by some modes of operation as an additional initial input.

AESB_IVR0 corresponds to the first word of the Initialization Vector, AESB_IVR3 to the last one.

These registers are write-only to prevent the Initialization Vector from being read by another application.

For CBC mode, the IV input value corresponds to the initialization vector.

For CTR mode, the IV input value corresponds to the initial counter value.

**Note:** These registers are not used in ECB mode and must not be written. For Automatic Bridge dedicated mode, the IV input value corresponds to the initial nonce.

# 59. Advanced Encryption Standard (AES)

## 59.1 Description

The Advanced Encryption Standard (AES) is compliant with the American FIPS (Federal Information Processing Standard) Publication 197 specification.

The AES supports the following confidentiality modes of operation for symmetrical key block cipher algorithms: ECB, CBC, OFB, CFB, CTR, and XTS), as specified in the NIST Special Publication 800-38A Recommendation, as well as Galois/Counter Mode (GCM) as specified in the NIST Special Publication 800-38D Recommendation. It is compatible with all these modes via DMA Controller channels, minimizing processor intervention for large buffer transfers.

The AES key is loaded by the software.

The 128-bit/192-bit/256-bit AES key is stored in the AES Key Register made of four/six/eight 32-bit write-only AES Key Word registers (AES_KEYWR0–7).

The 128-bit input data and initialization vector (for some modes) are each stored in four 32-bit write-only AES Input Data registers (AES_IDATAR0–3) and AES Initialization Vector registers (AES_IVR0–3).

As soon as the initialization vector, the input data and the key are configured, the encryption/decryption process may be started. Then the encrypted/decrypted data are ready to be read out on the four 32-bit AES Output Data registers (AES_ODATAR0–3) or through the DMA channels.

## 59.2 Embedded Characteristics

- Compliant with FIPS Publication 197, Advanced Encryption Standard (AES)
- 128-bit/192-bit/256-bit Cryptographic Key
- 10/12/14 Clock Cycles Encryption/Decryption Inherent Processing Time with a 128-bit/192-bit/256-bit Cryptographic Key
- Double Input Buffer Optimizes Runtime
- Automatic Padding supported for IPSEC and SSL standards
- IPSEC and SSL Protocol Layers Improved Performances (Tightly coupled with SHA)
- Support of the Modes of Operation Specified in the NIST Special Publication 800-38A and NIST Special Publication 800-38D:
  - Electronic Codebook (ECB)
  - Cipher Block Chaining (CBC) including CBC-MAC
  - Cipher Feedback (CFB)
  - Output Feedback (OFB)
  - Counter (CTR)
  - Galois/Counter Mode (GCM)
  - XEX-Based Tweaked-Codebook Mode (XTS)
- 8, 16, 32, 64 and 128-bit Data Sizes Possible in CFB Mode
- Last Output Data Mode Allows Optimized Message Authentication Code (MAC) Generation
- Connection to DMA Optimizes Data Transfers for all Operating Modes

## 59.3 Product Dependencies

### 59.3.1 Power Management

The AES is clocked through the Power Management Controller (PMC), so the programmer must first configure the PMC to enable the AES clock.

### 59.3.2 Interrupt Sources

The AES interface has an interrupt line connected to the Interrupt Controller.

Handling the AES interrupt requires programming the Interrupt Controller before configuring the AES.

## 59.4 Functional Description

The Advanced Encryption Standard (AES) specifies a FIPS-approved cryptographic algorithm that can be used to protect electronic data. The AES algorithm is a symmetric block cipher that can encrypt (encipher) and decrypt (decipher) information.

Encryption converts data to an unintelligible form called ciphertext. Decrypting the ciphertext converts the data back into its original form, called plaintext. The CIPHER bit in the AES Mode register (AES_MR) allows selection between the encryption and the decryption processes.

The AES is capable of using cryptographic keys of 128/192/256 bits to encrypt and decrypt data in blocks of 128 bits. This 128-bit/192-bit/256-bit key is defined in the user interface AES_KEYWRx register.

The input to the encryption processes of the CBC, CFB, and OFB modes includes, in addition to the plaintext, a 128-bit data block called the initialization vector (IV), which must be set in AES_IVRx. The initialization vector is used in an initial step in the encryption of a message and in the corresponding decryption of the message. AES_IVRx are also used by the CTR mode to set the counter value.

### 59.4.1 AES Register Endianness

In ARM processor-based products, the system bus and processors manipulate data in little-endian form. The AES interface requires little-endian format words. However, in accordance with the protocol of the FIPS 197 specification, data is collected, processed and stored by the AES algorithm in big-endian form.

The following example illustrates how to configure the AES:

If the first 64 bits of a message (according to FIPS 197, i.e., big-endian format) to be processed is 0xcafedeca_01234567, then AES_IDATAR0 and AES_IDATAR1 registers must be written with the following pattern:

- AES_IDATAR0 = 0xcadefeca
- AES_IDATAR1 = 0x67452301

### 59.4.2 Operating Modes

The AES supports the following modes of operation:

- ECB: Electronic Codebook
- CBC: Cipher Block Chaining
  - CBC-MAC: Useful for CMAC hardware acceleration
- OFB: Output Feedback
- CFB: Cipher Feedback

  - CFB8 (CFB where the length of the data segment is 8 bits)
  - CFB16 (CFB where the length of the data segment is 16 bits)
  - CFB32 (CFB where the length of the data segment is 32 bits)
  - CFB64 (CFB where the length of the data segment is 64 bits)
  - CFB128 (CFB where the length of the data segment is 128 bits)
- CTR: Counter
- GCM: Galois/Counter Mode
- XTS: XEX-based Tweaked-codebook Mode

The data preprocessing, data postprocessing and data chaining for the concerned modes are performed automatically. Refer to the NIST Special Publication 800-38A and NIST Special Publication 800-38D for more complete information.

Mode selection is done by configuring the OPMOD field in AES_MR.

In CFB mode, five data sizes are possible (8, 16, 32, 64 or 128 bits), configurable by means of AES_MR.CFBS.

In CTR mode, the size of the block counter embedded in the module is 16 bits. Therefore, there is a rollover after processing 1 Mbyte of data. If the file to be processed is greater than 1 Mbyte, this file must be split into fragments of 1 Mbyte or less for the first fragment if the initial value of the counter is greater than 0. Prior to loading the first fragment into AES_IDATARx, AES_IVRx must be fully programmed with the initial counter value. For any fragment, after the transfer is completed and prior to transferring the next fragment, AES_IVRx must be programmed with the appropriate counter value.

### 59.4.3 Last Output Data Mode (CBC-MAC)

This mode is used to generate cryptographic checksums on data (MAC) by means of cipher block chaining encryption algorithm (CBC-MAC algorithm for example).

The CMAC algorithm is a variant of CBC-MAC with post-processing requiring one-block encryption in ECB mode. Thus CBC-MAC is useful to accelerate CMAC.

After each end of encryption/decryption, the output data are available either on AES_ODATARx for Manual and Auto mode, or at the address specified in the receive buffer pointer for DMA mode (see the table "Last Output Data Mode Behavior versus Start Modes").

AES_MR.LOD allows retrieval of only the last data of several encryption/decryption processes.

Therefore, there is no need to define a read buffer in DMA mode.

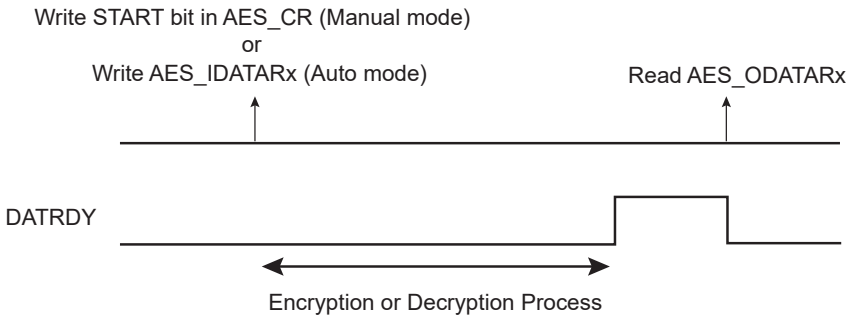
This data are only available in AES_ODATARx.

#### 59.4.3.1 Manual and Auto Modes

##### 59.4.3.1.1 If AES_MR.LOD = 0

The DATRDY flag is cleared when at least one AES_ODATARx is read (see the figure below).

**Figure 59-1. Manual and Auto Modes with AES_MR.LOD = 0**



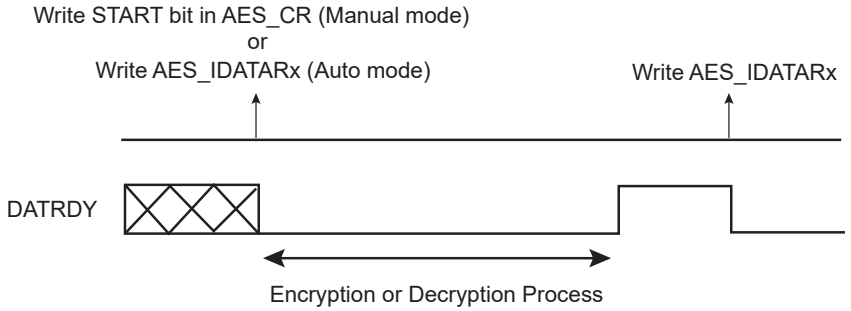

If the user does not want to read AES_ODATARx between each encryption/decryption, the DATRDY flag will not be cleared. If the DATRDY flag is not cleared, the user cannot know the end of the following encryptions/decryptions.

#### 59.4.3.1.2 If AES_MR.LOD = 1

This mode is optimized to process AES CBC-MAC operating mode.

The DATRDY flag is cleared when at least one AES_IDATAR is written (see the figure below). No additional AES_ODATAR reads are necessary between consecutive encryptions/decryptions.

**Figure 59-2. Manual and Auto Modes with AES_MR.LOD = 1**



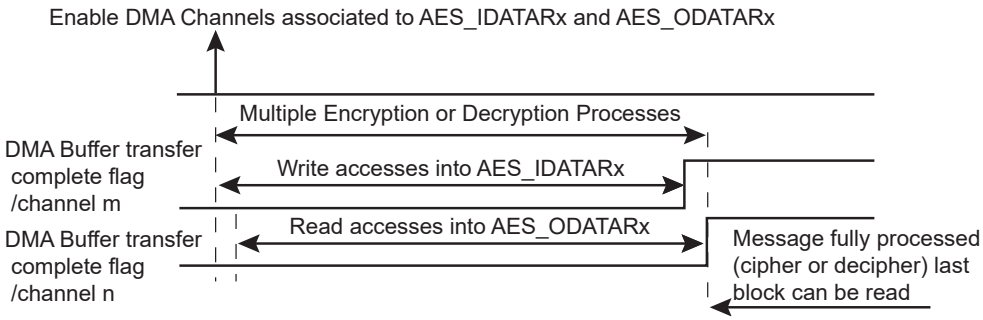

### 59.4.3.2 DMA Mode

#### 59.4.3.2.1 If AES_MR.LOD = 0

This mode may be used for all AES operating modes except CBC-MAC where AES_MR.LOD = 1 mode is recommended.

The end of the encryption/decryption is indicated by the end of DMA transfer associated to AES_ODATARx (see the figure below). Two DMA channels are required: one for writing message blocks to AES_IDATARx and one to obtain the result from AES_ODATARx.

**Figure 59-3. DMA Transfer with AES_MR.LOD = 0**



#### 59.4.3.2.2 If AES_MR.LOD = 1

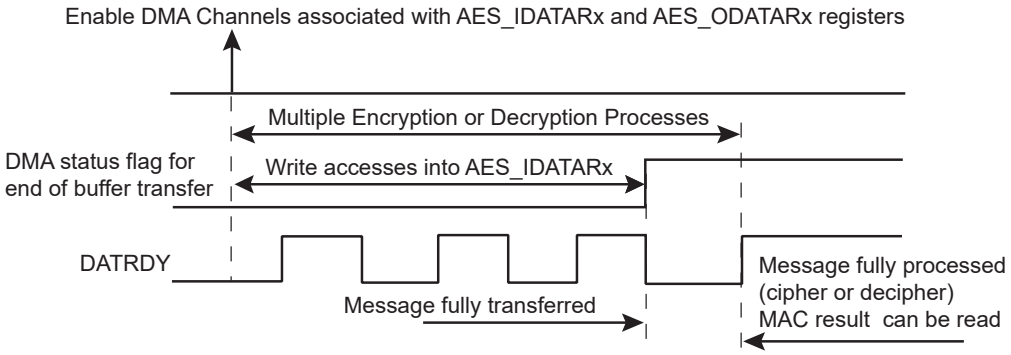
This mode is optimized to process AES CBC-MAC operating mode.

The user must first wait for the DMA buffer transfer complete flag, then for the flag DATRDY to rise to ensure that the encryption/decryption is completed (see the figure below).

The DMA receive channel must not be used. Prior to reading the CBC-MAC result, AES_MR.SMOD must be written to ‘0’. To restart a CBC-MAC on a new buffer, AES_MR.SMOD must be written to ‘2’.

The output data are only available on AES_ODATARx.

**Figure 59-4. DMA Transfer with AES_MR.LOD = 1**




The table below summarizes the different cases.

**Table 59-1. Last Output Data Mode Behavior versus Start Modes**

| Sequence | Manual and Auto Modes | | DMA Transfer | |
|---|---|---|---|---|
| | AES_MR.LOD = 0 | AES_MR.LOD = 1 | AES_MR.LOD = 0 | AES_MR.LOD = 1 |
| DATRDY Flag Clearing Condition(1) | At least one AES_ODATAR must be read | At least one AES_IDATAR must be written | Not used | Managed by the DMA |
| End of Encryption/ Decryption Notification | DATRDY | DATRDY | 2 DMA Buffer transfer complete flags (channel m and channel n) | DMA buffer transfer complete flag, then AES DATRDY flag |
| Encrypted/ Decrypted Data Result Location | In AES_ODATARx | In AES_ODATARx | At the address specified in the Channel Buffer Transfer Descriptor | In AES_ODATARx |

**Note:**

1. Depending on the mode, there are other ways of clearing the DATRDY flag. See AES Interrupt Status Register.

**WARNING** In DMA mode, reading AES_ODATARx before the last data transfer may lead to unpredictable results.

### 59.4.4 Galois/Counter Mode (GCM)

#### 59.4.4.1 Description

GCM comprises the AES engine in CTR mode along with a universal hash function (GHASH engine) that is defined over a binary Galois field to produce a message authentication tag (the AES CTR engine and the GHASH engine are depicted in the figure below.

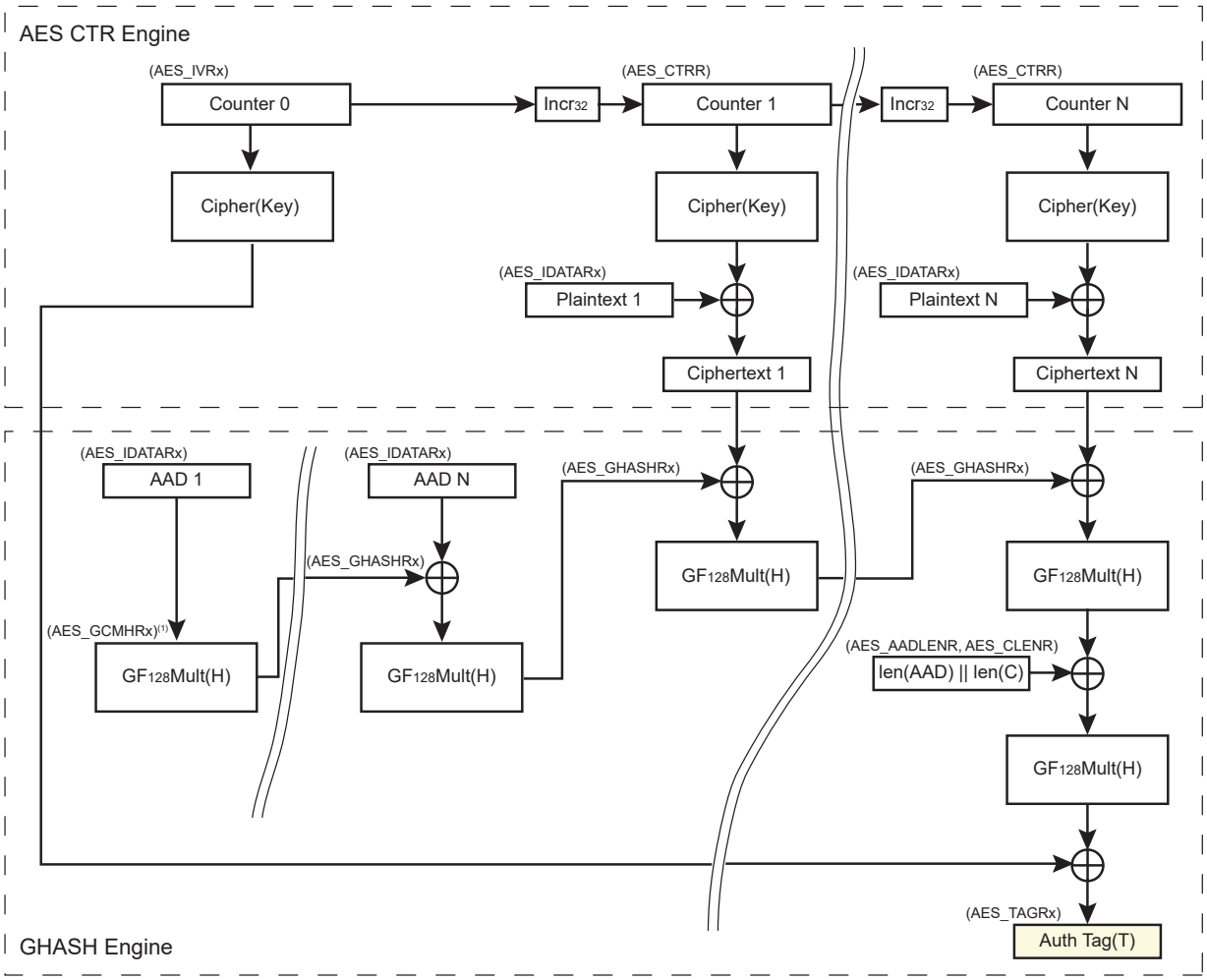
The GHASH engine processes data packets after the AES operation. GCM assures the confidentiality of data through the AES Counter mode of operation for encryption. Authenticity of the confidential data is assured through the GHASH engine. GCM can also provide assurance of data that is not encrypted. Refer to the NIST Special Publication 800-38D for more complete information.

GCM can be used with or without the DMA master. Messages may be processed as a single complete packet of data or they may be broken into multiple packets of data over time.

GCM processing is computed on 128-bit input data fields. There is no support for unaligned data. The AES key length can be whatever length is supported by the AES module.

The recommended programming procedure when using DMAPDC is described in the section GCM Processing.

**Figure 59-5. GCM Block Diagram**




Note: 1. Optional

#### 59.4.4.2 Key Writing and Automatic Hash Subkey Calculation

Whenever a new key is written to the hardware, two automatic actions are processed:

- GCM Hash Subkey *H* generation—The GCM hash subkey (*H*) is automatically generated. The GCM hash subkey generation must be complete before doing any other action. AES_ISR.DATRDY indicates when the subkey generation is complete (with interrupt if configured). The GCM hash subkey calculation is processed with the formula *H* = CIPHER(Key, <128 bits to zero>). The generated GCM *H* value is then available in AES_GCMHRx. If the application software requires a specific hash subkey, the automatically generated *H* value can be overwritten in AES_GCMHRx. AES_GCMHRx can be written after the end of the hash subkey generation (see AES_ISR.DATRDY) and prior to starting the input data feed.
- AES_GHASHRx Clear—AES_GHASHRx are automatically cleared. If a hash initial value is needed for the GHASH, it must be written to AES_GHASHRx
  - after a write to the AES Key Register, if any
  - before starting the input data feed

#### 59.4.4.3 GCM Processing

GCM processing is made up of three phases:

1. Processing the Additional Authenticated Data (*AAD*), hash computation only.
2. Processing the Ciphertext (C), hash computation + ciphering/deciphering.
3. Generating the Tag using length of *AAD*, length of C and $J_0$ (refer to NIST documentation for details).

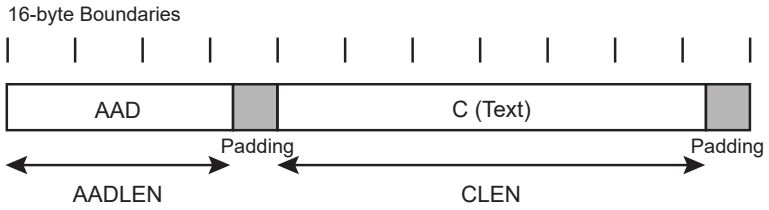
The Tag generation can be done either automatically, after the end of *AAD*/C processing if AES_MR.GTAGEN is set, or manually using AES_GHASHRx.GHASH (see subsections Processing a Complete Message with Tag Generation and Manual GCM Tag Generation for details).

##### 59.4.4.3.1 Processing a Complete Message with Tag Generation

Use this procedure only if J0 four LSB bytes ≠ 0xFFFFFFFF.

**Note:** If $J_0$ four LSB bytes = 0xFFFFFFFF or if the value is unknown, use the procedure described in Processing a Complete Message without Tag Generation followed by the procedure in Manual GCM Tag Generation.

**Figure 59-6. Full Message Alignment**




To process a complete message with Tag generation, the sequence is as follows:

1. Set AES_MR.OPMOD to GCM and AES_MR.GTAGEN to '1'.
2. Set the AES Key Register and wait until AES_ISR.DATRDY is set (GCM hash subkey generation complete); use interrupt if needed. See Key Writing and Automatic Hash Subkey Calculation.
3. Calculate the $J_0$ value as described in NIST documentation $J_0$ = IV || $0^{31}$ || 1 when len(IV) = 96 and $J_0$ = $GHASH_H$(IV || $0^{s+64}$ || [len(IV)]64) if len(IV) ≠ 96. See Processing a Message with only AAD (GHASHH) for $J_0$ generation.
4. Set AES_IVRx.IV with inc32($J_0$) ($J_0$ + 1 on 32 bits).
5. Configure AES_AADLENR.AADLEN and AES_CLENR.CLEN.

6. Fill AES_IDATARx.IDATA with the message to process according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).
7. Wait for TAGRDY to be set (use interrupt if needed), then read AES_TAGRx.TAG to obtain the authentication tag of the message.

#### 59.4.4.3.2 Processing a Complete Message without Tag Generation

Processing a message without generating the Tag can be used to customize the Tag generation, or to process a fragmented message. To manually generate the GCM Tag, see Manual GCM Tag Generation.

To process a complete message without Tag generation, the sequence is as follows:

1. Set AES_MR.OPMOD to GCM and AES_MR.GTAGEN to '0'.
2. Set the AES Key Register and wait until AES_ISR.DATRDY is set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in AES_GCMHRx. See Key Writing and Automatic Hash Subkey Calculation.
3. Calculate the J0 value as described in NIST documentation $J_0$ = IV || $0^{31}$ || 1 when len(IV) = 96 and $J_0$ = $GHASH_H$(IV || $0^{s+64}$ || [len(IV)]64) if len(IV) ≠ 96. See Processing a Message with only AAD (GHASHH) for $J_0$ generation example when len(IV) ≠ 96.
4. Set AES_IVRx.IV with inc32($J_0$) ($J_0$ + 1 on 32 bits).
5. Configure AES_AADLENR.AADLEN and AES_CLENR.CLEN.
6. Fill AES_IDATARx.IDATA with the message to process according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).
7. Make sure the last output data have been read if AES_CLENR.CLEN ≠ 0 (or wait for DATRDY), then read AES_GHASHRx.GHASH to obtain the hash value after the last processed data.

#### 59.4.4.3.3 Processing a Fragmented Message without Tag Generation

If needed, a message can be processed by fragments, in such case automatic GCM Tag generation is not supported.

To process a message by fragments, the sequence is as follows:

- First fragment:

1. Set AES_MR.OPMOD to GCM and AES_MR.GTAGEN to '0'.
2. Set the AES Key Register and wait for AES_ISR.DATRDY to be set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in AES_GCMHRx. See Key Writing and Automatic Hash Subkey Calculation.
3. Calculate the $J_0$ value as described in NIST documentation $J_0$ = IV || $0^{31}$ || 1 when len(IV) = 96 and $J_0$ = $GHASH_H$(IV || $0^{s+64}$ || [len(IV)]64) if len(IV) ≠ 96. See Processing a Message with only AAD (GHASHH) for $J_0$ generation example when len(IV) ≠ 96.
4. Set AES_IVRx.IV with inc32($J_0$) ($J_0$ + 1 on 32 bits).
5. Configure AES_AADLENR.AADLEN and AES_CLENR.CLEN according to the length of the first fragment, or set the fields with the full message length (both configurations work).
6. Fill AES_IDATARx.IDATA with the first fragment of the message to process (aligned on 16-byte boundary) according to the SMOD configuration used. If Manual Mode or Auto Mode is used the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).

7. Make sure the last output data have been read if the fragment ends in C phase (or wait for DATRDY if the fragment ends in *AAD* phase), then read AES_GHASHRx.GHASH to obtain the value of the hash after the last processed data and finally read AES_CTR.CTR to obtain the value of the CTR encryption counter (not needed when the fragment ends in AAD phase).

- Next fragment (or last fragment):

1. Set AES_MR.OPMOD to GCM and AES_MR.GTAGEN to '0'.
2. Set the AES Key Register and wait until AES_ISR.DATRDY is set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in AES_GCMHRx. See Key Writing and Automatic Hash Subkey Calculation.
3. Set AES_IVRx.IV as follows:
   - If the first block of the fragment is a block of Additional Authenticated data, set AES_IVRx.IV with the $J_0$ initial value
   - If the first block of the fragment is a block of Plaintext data, set AES_IVRx.IV with a value constructed as follows: 'LSB96($J_0$) || CTR' value, (96 bit LSB of $J_0$ concatenated with saved CTR value from previous fragment).
4. Configure AES_AADLENR.AADLEN and AES_CLENR.CLEN according to the length of the current fragment, or set the fields with the remaining message length, both configurations work.
5. Fill AES_GHASHRx.GHASH with the value stored after the previous fragment.
6. Fill AES_IDATARx.IDATA with the current fragment of the message to process (aligned on 16 byte boundary) according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).
7. Make sure the last output data have been read if the fragment ends in C phase (or wait for DATRDY if the fragment ends in AAD phase), then read AES_GHASHRx.GHASH to obtain the value of the hash after the last processed data and finally read AES_CTR.CTR to obtain the value of the CTR encryption counter (not needed when the fragment ends in *AAD* phase).

**Note:** Step 1 and 2 are required only if the value of the concerned registers has been modified.

Once the last fragment has been processed, the GHASH value will allow manual generation of the GCM tag. See Manual GCM Tag Generation.

#### 59.4.4.3.4 Manual GCM Tag Generation

This section describes the last steps of the GCM Tag generation.

The Manual GCM Tag Generation is used to complete the GCM Tag Generation when the message has been processed without Tag Generation.

**Note:** The Message Processing without Tag Generation must be finished before processing the Manual GCM Tag Generation.

To generate a GCM Tag manually, the sequence is as follows:

Processing S = $GHASH_H$ (*AAD* || 0v || C || 0u || [len(*AAD*)]64 || [len(C)]64):

1. Set AES_MR.OPMOD to GCM and AES_MR.GTAGEN to '0'.
2. Set the AES Key Register and wait for AES_ISR.DATRDY to be set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in AES_GCMHRx. See Key Writing and Automatic Hash Subkey Calculation.

3. Configure AES_AADLENR.AADLEN to 0x10 (16 bytes) and AES_CLENR.CLEN to '0'. This will allow running a single $GHASH_H$ on a 16-byte input data (see the figure below).
4. Fill AES_GHASHRx.GHASH with the state of the GHASH field stored at the end of the message processing.
5. Fill AES_IDATARx.IDATA according to the SMOD configuration used with 'len(*AAD*)64 || len(*C*)64' value as described in the NIST documentation and wait for DATRDY to be set; use interrupt if needed.
6. Read AES_GHASHRx.GHASH to obtain the current value of the hash.

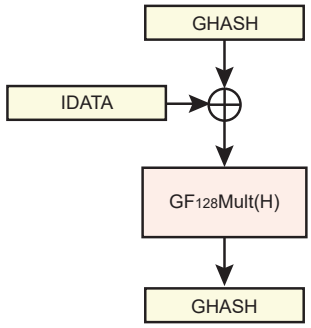
Processing $T$ = GCTRK($J_0$, S):

1. Set AES_MR.OPMOD to CTR.
2. Set AES_IVRx.IV with '$J_0$' value.
3. Fill AES_IDATARx.IDATA with the GHASH value read at step 6 and wait for DATRDY to be set (use interrupt if needed).
4. Read AES_ODATARx.ODATA to obtain the GCM Tag value.

**Note:** Step 4 is optional if the GHASH field is to be filled with value '0' (0 length packet for instance).

#### 59.4.4.3.5 Processing a Message with only AAD (GHASHH)

**Figure 59-7. Single $GHASH_H$ Block Diagram (AADLEN ≤ 0x10 and CLEN = 0)**




It is possible to process a message with only *AAD* setting the CLEN field to '0' in AES_CLENR, this can be used for J0 generation when len(*IV*) ≠ 96 for instance.

Example: Processing $J_0$ when len(*IV*) ≠ 96

To process $J_0 = GHASH_H(IV \,||\, 0^{s+64} \,||\, [len(IV)]64)$, the sequence is as follows:

1. Set AES_MR.OPMOD to GCM and AES_MR.GTAGEN to '0'.

1. Set the AES Key Register and wait until AES_ISR.DATRDY is set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in AES_GCMHRx. See Key Writing and Automatic Hash Subkey Calculation.
2. Configure AES_AADLENR.AADLEN with 'len($IV \,||\, 0^{s+64} \,||\, [len(IV)]64$)' in and AES_CLENR.CLEN to '0'. This will allow running a $GHASH_H$ only.
3. Fill AES_IDATARx.IDATA with the message to process ($IV \,||\, 0^{s+64} \,||\, [len(IV)]64$) according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when a $GHASH_H$ step is over (use interrupt if needed).
4. Read AES_GHASHRx.GHASH to obtain the $J_0$ value.
   Note: The GHASH value can be overwritten at any time by writing the value of AES_GHASHRx.GHASH, used to perform a $GHASH_H$ with an initial value for GHASH (write GHASH field between step 3 and step 4 in this case).

#### 59.4.4.3.6 Processing a Single GF128 Multiplication

The AES can also be used to process a single multiplication in the Galois field on 128 bits ($GF_{128}$) using a single $GHASH_H$ with custom *H* value (see the figure above).

To run a $GF_{128}$ multiplication (A x B), the sequence is as follows:

1. Set AES_MR.OPMOD to GCM and AES_MR.GTAGEN to '0'.

1. Configure AES_AADLENR.AADLEN with 0x10 (16 bytes) and AES_CLENR.CLEN to '0'. This will allow running a single $GHASH_H$.
2. Fill AES_GCMHRx.H with B value.
3. Fill AES_IDATARx.IDATA with the A value according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when a GHASHH computation is over (use interrupt if needed).
4. Read AES_GHASHRx.GHASH to obtain the result.

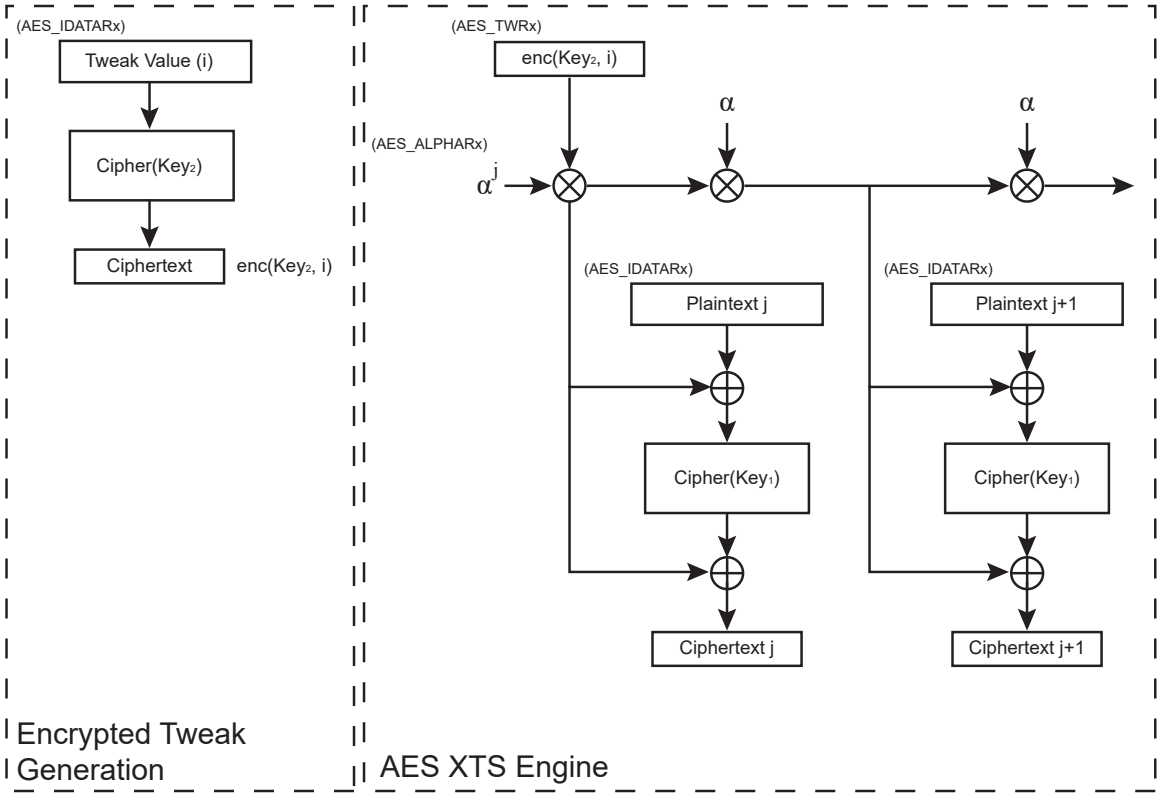
**Note:** AES_GHASHRx.GHASH can be initialized with a value C between step 3 and step 4 to run a ((A XOR C) x B) $GF_{128}$ multiplication.

### 59.4.5 XEX-based Tweaked-codebook Mode (XTS)

XTS mode comprises the AES engine with XOR on inputs and outputs. After each encryption/decryption, the value used for the XOR is multiplied by the first GF($2^{128}$) alpha primitive (0x2) and then used for the next encryption/decryption. The XTS mode uses two different keys and defines a Tweak Value (i) as additional input.

XTS processing is computed on 128-bit input data fields. There is no support for unaligned data (padding must be done manually if needed). The AES key length can be any length supported by the AES module.

**Figure 59-8. XTS Block Diagram**



#### 59.4.5.1 XTS Processing Procedure

XTS processing comprises two phases:

1. Generate encrypted tweak with Key2 (this step is only required for the first processing, further consecutive processing does not require this step).
2. Process the data giving encrypted tweak and first alpha primitive for the first encryption/decryption.

##### 59.4.5.1.1 Encrypted Tweak Generation

In the case of a new encryption/decryption, it is necessary to first encrypt the Tweak Value (i) with Key2. Here are the steps to follow to perform this step:

1. Set AES_MR.OPMODE to ECB and AES_MR.CIPHER to '1'.
2. Set the AES Key Register with Key2.
3. Fill AES_IDATARx.IDATA with the Tweak value (i) according to the SMOD configuration used. If Manual mode or Auto mode is used, the DATRDY bit indicates when the data have been processed and can be read in AES_ODATARx.

##### 59.4.5.1.2 Data Processing

To process data using XTS mode, follow the steps below:

1. Set AES_MR.OPMODE to XTS.
2. Set the AES Key Register with Key1.
3. Only if the data to process is the first to be processed in the data unit, or if the data block to process is not consecutive to the previous processed data block in the same data unit, then two additional mandatory steps are required:
   a. AES_TWRx must be written with the encrypted Tweak Value (see Encrypted Tweak Generation for details) with bytes swapped as described in AES Register Endianness.

   b. Write AES_ALPHARx with the alpha primitive corresponding to the block number in the data unit.
4. Fill AES_IDATARx.IDATA with the data to process according to the SMOD configuration used. If Manual mode or Auto mode is used, the DATRDY bit indicates when the data have been processed and can be read in AES_ODATARx. Repeat Step 4 as long as consecutive data blocks are processed in the same data unit.

### 59.4.6 Double Input Buffer

AES_IDATARx can be double-buffered to reduce the runtime of large files.

This mode allows a new message block to be written when the previous message block is being processed. This is only possible when DMA accesses are performed (AES_MR.SMOD = 2).

AES_MR.DUALBUFF must be set to '1' to access the double buffer.

### 59.4.7 Start Modes

AES_MR.SMOD allows selection of the encryption (or decryption) Start mode.

#### 59.4.7.1 Manual Mode

The sequence of actions is as follows:

1. Write AES_MR with all required fields, including but not limited to SMOD and OPMOD.
2. Write the 128-bit/192-bit/256-bit AES key in AES_KEYWRx.
3. Write the initialization vector (or counter) in AES_IVRx.
   Note: AES_IVRx concerns all modes except ECB.
4. Set the bit DATRDY (Data Ready) in the AES Interrupt Enable register (AES_IER), depending on whether an interrupt is required or not at the end of processing.

5. Write the data to be encrypted/decrypted in the authorized AES_IDATARx (see the table below).
6. Set the START bit in the AES Control register (AES_CR) to begin the encryption or the decryption process.
7. When processing completes, the DATRDY flag in the AES Interrupt Status register (AES_ISR) is raised. If an interrupt has been enabled by setting AES_IER.DATRDY, the interrupt line of the AES is activated.
8. When software reads one of AES_ODATARx, AES_IER.DATRDY is automatically cleared.

**Table 59-2. Authorized Input Data Registers**

| Operating Mode | Input Data Registers to Write |
|---|---|
| ECB | All |
| CBC | All |
| OFB | All |
| 128-bit CFB | All |
| 64-bit CFB | AES_IDATAR0 and AES_IDATAR1 |
| 32-bit CFB | AES_IDATAR0 |
| 16-bit CFB | AES_IDATAR0 |
| 8-bit CFB | AES_IDATAR0 |
| CTR | All |
| GCM | All |
| XTS | All |

**Note:**

1. In 64-bit CFB mode, writing to AES_IDATAR2 and AES_IDATAR3 is not allowed and may lead to errors in processing.
2. In 32, 16, and 8-bit CFB modes, writing to AES_IDATAR1, AES_IDATAR2 and AES_IDATAR3 is not allowed and may lead to errors in processing.

#### 59.4.7.2 Auto Mode

The Auto Mode is similar to the manual one, except that in this mode, as soon as the correct number of AES_IDATARx is written, processing is automatically started without any action in AES_CR.

#### 59.4.7.3 DMA Mode

The DMA Controller can be used in association with the AES to perform an encryption/decryption of a buffer without any action by software during processing.

AES_MR.SMOD must be configured to 2 and the DMA must be configured with non-incremental addresses.

For all operating modes except CBC-MAC (AES_MR.LOD=1), 2 DMA channels must be programmed (transmit and receive). In CBC-MAC, only 1 transmit channel must be programmed.

The start address of any transfer descriptor must be configured with the address of AES_IDATAR0.

The DMA chunk size configuration depends on the AES mode of operation and is listed in the table below.

When writing data to AES with a first DMA channel, data are first fetched from a memory buffer (source data). It is recommended to configure the size of source data to “words” even for CFB modes. On the contrary, the destination data size depends on the mode of operation. When reading data from the AES with the second DMA channel, the source data is the data read from AES and data destination is the memory buffer. In this case, the source data size depends on the AES mode of operation and is listed in the table below.

**Table 59-3. DMA Data Transfer Type for the Different Operating Modes**

| Operating Mode | Chunk Size | Destination/Source Data Transfer Type |
|---|---|---|
| ECB | 4 | Word |
| CBC | 4 | Word |
| OFB | 4 | Word |
| CFB 128-bit | 4 | Word |
| CFB 64-bit | 1 | Word |
| CFB 32-bit | 1 | Word |
| CFB 16-bit | 1 | Half-word |
| CFB 8-bit | 1 | Byte |
| CTR | 4 | Word |
| GCM | 4 | Word |
| XTS | 4 | Word |

### 59.4.8 Automatic Padding Mode

When Automatic Padding mode is configured, the message is automatically padded after the last block is written. Depending on the size of the message, either a padding is performed after the last part of the message and padding blocks are added, or only padding blocks are added.

IPSEC and SSL padding standards are both supported.

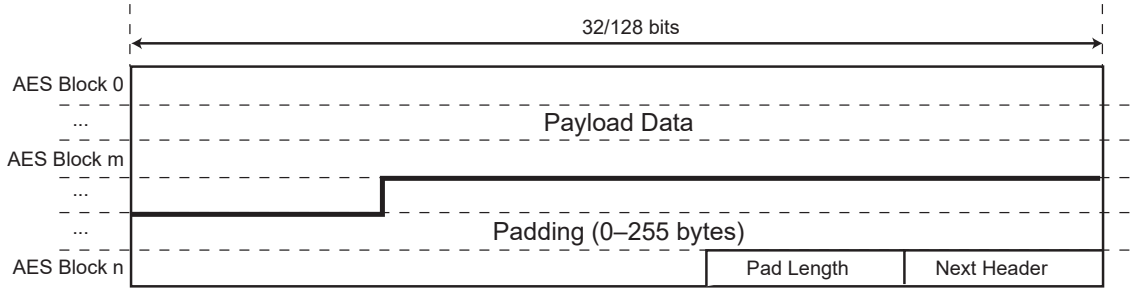
The auto padding feature only supports CBC and CTR modes.

**Note:** When automatic padding is enabled and AES_MR.SMOD=2, AES_MR.DUALBUFF must be cleared.

#### 59.4.8.1 IPSEC Padding

Automatic Padding is enabled by writing a ‘1’ to the APEN bit in the AES Extended Mode register (AES_EMR).IPSEC padding mode is selected by writing a ‘0’ to AES_EMR.APM.

**Figure 59-9. IPSEC Padding**




Each byte of the padding area contains incremental integer values.

The “Pad Length” in bytes is configured in AES_EMR.PADLEN and the “Next Header” value is configured in AES_EMR.NHEAD. AES_EMR.PADLEN must be configured with the length of the padding section, not including the length of the “Pad Length” and “Next Header” sections.

The BCNT field in the AES Byte Counter register (AES_BCNT) defines the length, in bytes, of the message to process. It must be configured before writing the first data in AES_IDATARx and the remaining bytes to process can be read at anytime (BCNT value is decremented after each AES_IDATARx access).

AES_BCNT.BCNT and AES_EMR.PADLEN must be configured so that the sum of the length of the message (Payload Data) and of the length of the Padding, Pad Length (1 byte) and Next Header (1 byte) sections is a multiple of the AES block size (128 bits).

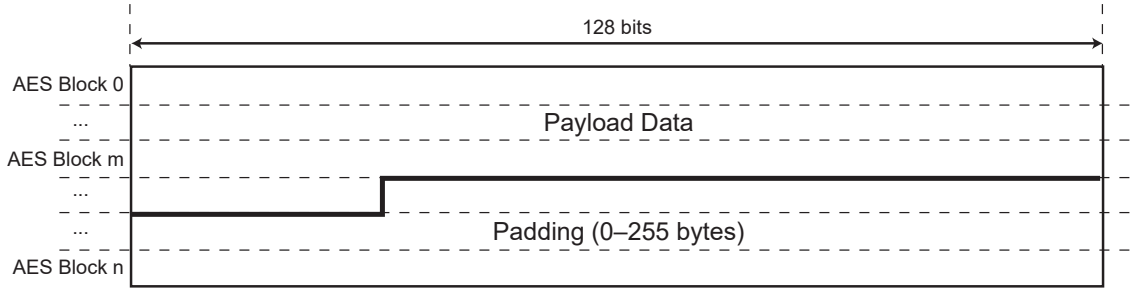
To process an IPSEC message using auto-padding, the sequence is as follows:

1. Set AES_MR.OPMOD to either CBC or CTR mode.
2. Set AES_EMR.APEN to ‘1’, AES_EMR.APM to ‘0’, AES_EMR.PADLEN to the desired padding length in byte and AES_EMR.NHEAD to the desired Next Header field value.
3. Configure AES_BCNT.BCNT with the whole message length, without padding, in byte.
4. Set the AES Key Register.
5. Set AES_IVRx.IV if needed.
6. Fill AES_IDATARx.IDATA with the message to process according to the SMOD configuration used. On the last data block, write only what is necessary (e.g., write only AES_IDATAR0 if last block size is ≤ 32 bits).
7. Wait for the DATRDY flag to be raised, meaning auto-padding completion and last block processing.

#### 59.4.8.2 SSL Padding

Auto Padding is enabled by writing a ‘1’ to AES_EMR.APEN and SSL padding mode is selected by writing a ‘1’ to AES_EMR.APM.

**Figure 59-10. SSL Padding**




Each byte of the padding area contains the padding length.

The padding length is configured in AES_EMR.PADLEN.

AES_BCNT.BCNT defines the length, in bytes, of the message to process. It must be configured before writing the first data in AES_IDATARx and the remaining bytes to process can be read at anytime (BCNT value is decremented after each AES_IDATARx access).

AES_BCNT.BCNT and AES_EMR.PADLEN must be configured so that the length of the message plus the length of the padding section is a multiple of the AES block size (128 bits).

To process a complete SSL message, the sequence is as follows:

1. Set AES_MR.OPMOD to either CBC or CTR mode.
2. Set AES_EMR.APEN to ‘1’, AES_EMR.APM to ‘1’, AES_EMR.PADLEN to the desired padding length in bytes.
3. Set AES_BCNT.BCNT with the whole message length, without padding, in bytes.
4. Set the AES Key Register.
5. Set AES_IVRx.IV if needed.
6. Fill AES_IDATARx.IDATA with the message to process according to the SMOD configuration used. On the last data block write only what is necessary (e.g., write only AES_IDATAR0 if last block size is ≤ 32 bits).
7. Wait for the DATRDY flag to be raised, meaning auto-padding completion and last block processing.

#### 59.4.8.3 Flags

AES_ISR.EOPAD rises as soon as the automatic padding phase is over, meaning that all the extra padding blocks have been processed. Reading AES_ISR clears this flag.

AES_ISR.PLENERR indicates an error in the frame configuration, meaning that the whole message length including padding does not respect the standard selected. AES_ISR.PLENERR rises at the end of the frame in case of wrong message length and is cleared reading AES_ ISR.

In IPSEC/SSL standard message length including padding must be a multiple of the AES block size when CBC mode is used and multiple of 32-bit if CTR mode is used.

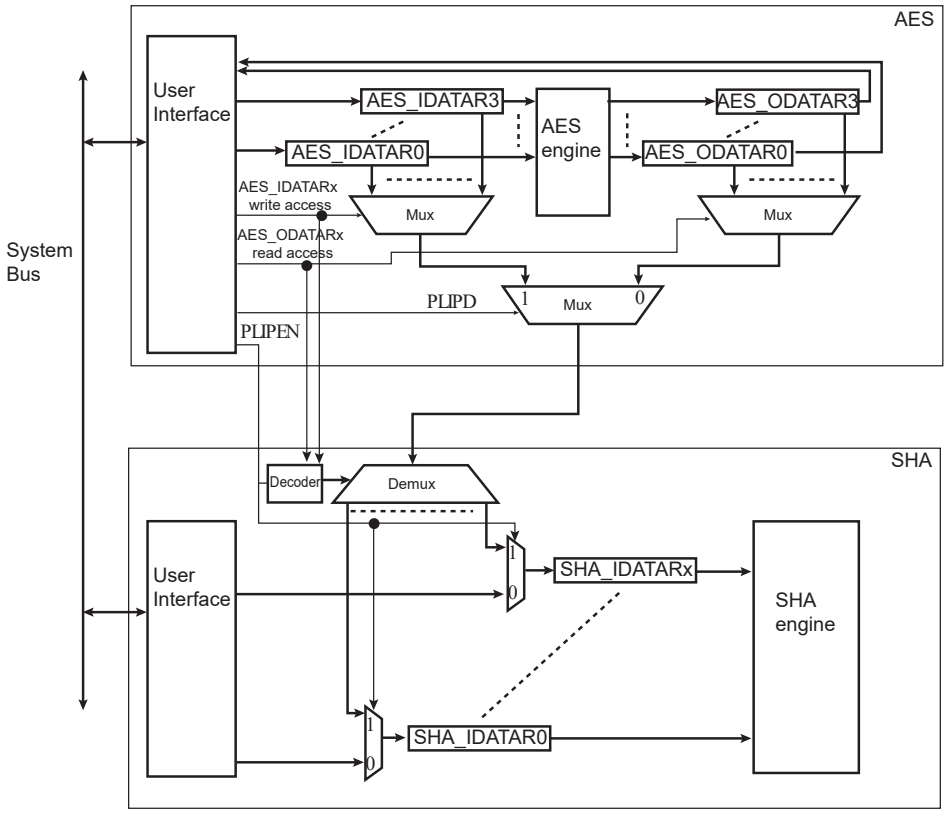
### 59.4.9 Secure Protocol Layers Improved Performances

Secure protocol layers such as IPSec require encryption and authentication. For IPSec, the authentication is based on HMAC, thus SHA is required. To optimize performance, the AES embeds a mode of operation that enables the SHA module to process the input or output data of the AES module. If this mode is enabled, write access is required only into AES_IDATARx registers, since SHA input data registers are automatically written by AES without software intervention. When the DMA is configured to transfer a buffer of data (input frame), only one transfer descriptor is required for both authentication and

encryption/decryption processes and only one buffer is transferred through the system bus (reducing the load of the system bus).

Improved performance for secure protocol layers requires AES_EMR.PLIPEN to be set.

**Figure 59-11. Secure Protocol Layers Improved Performances Block Diagram**




### 59.4.9.1 Cipher Mode

When AES_EMR.PLIPD is cleared and AES_EMR.PLIPEN=1, the message written into AES_IDATARx is first encrypted with the AES module and the encrypted message is authenticated with the SHA module. Therefore, when AES_EMR.PLIPD is cleared, AES_ODATARx are selected and sent to SHA_IDATARx as soon as AES_ODATARx are read. A read access in AES corresponds to a write access to the corresponding SHA_IDATARx. The number of SHA_IDATARx is greater than the number of AES_ODATARx, but the SHA module embeds the decoding logic to automatically dispatch AES_ODATARx values into the corresponding SHA_IDATARx without software intervention.

### 59.4.9.2 Decipher Mode

When AES_EMR.PLIPD is written to ‘1’ and AES_EMR.PLIPEN=1, the message written into AES_IDATARx is decrypted with the AES module and also sent to SHA for authentication. Therefore, when AES_EMR.PLIPD=1, AES_IDATARx are selected and sent to SHA_IDATARx as soon as AES_IDATARx are written. A write access in AES corresponds to a write access to the corresponding SHA_IDATARx. The number of SHA_IDATARx is greater than the number of AES_ODATARx, but the SHA module embeds the decoding logic to automatically dispatch AES_IDATARx values into the corresponding SHA_IDATARx without software intervention.

### 59.4.9.3 Encapsulating Security Payload (ESP) IPSec Examples

The following examples describe how to configure AES and SHA to optimize processing an ESP IPSec frame for maximum performance.

The cipher (or decipher) of an ESP IPSec frame requires both encryption (or decryption) and authentication.

For cipher, the input frame located in the system memory must first be padded and the resulting buffer encrypted. The encrypted frame must be written back to the system memory and sent to the authentication module.

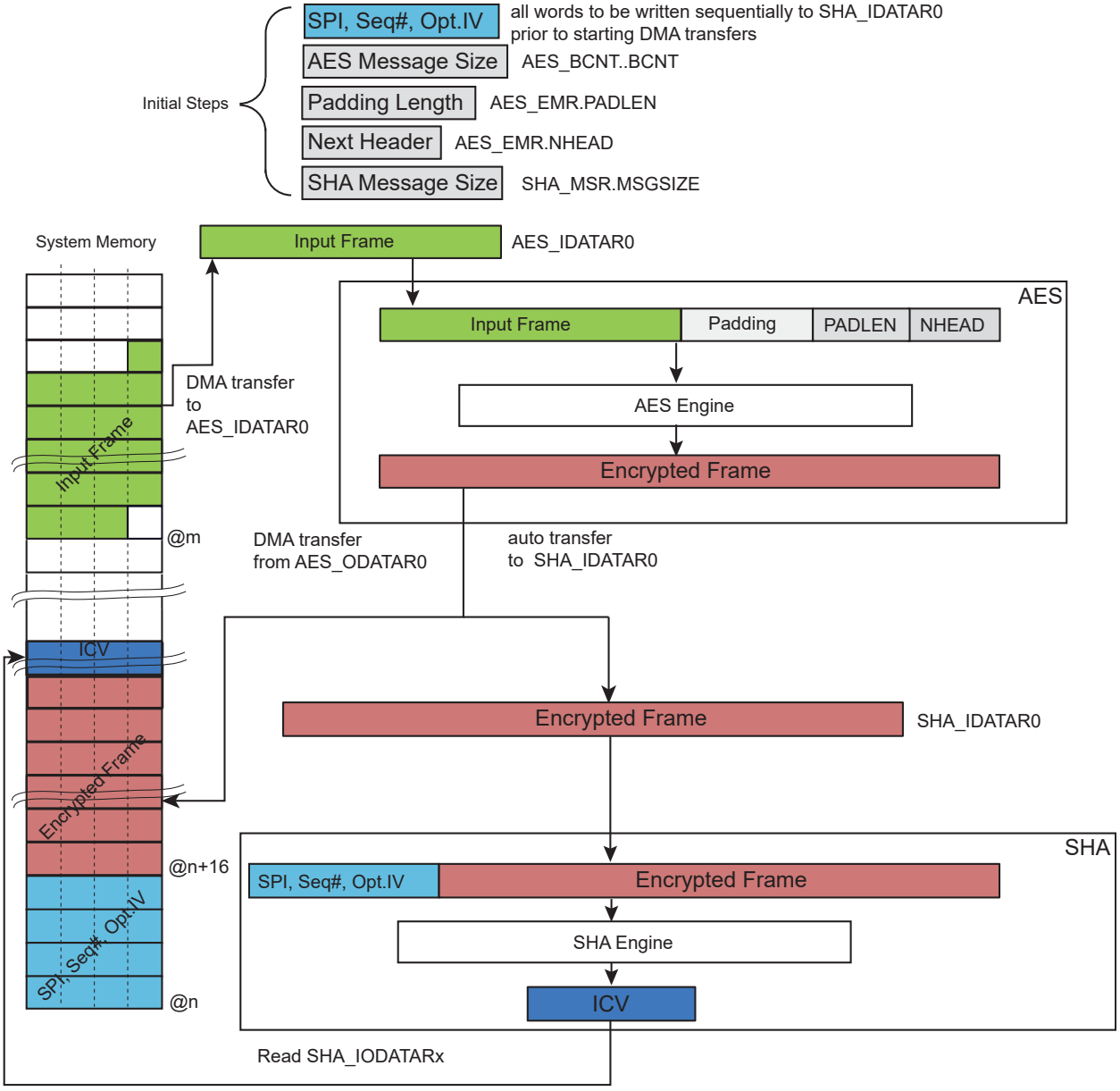
When the AES module is configured to improve the performance of the secure protocol layers (AES_EMR.PLIPEN = 1), the data transfers are simplified, limiting the bandwidth requirements on the system bus.

Before configuring the DMA to start the transfer of the data buffer (input frame) to the AES, the following actions must be taken in registers:

- AES_BCNT.BCNT must be configured with the length of the message (Input Frame).
- The padding length of the AES must be configured in AES_EMR.PADLEN. See Automatic Padding Mode for details.
- The next header value must be configured in AES_EMR.NHEAD.
- AES_MR.SMOD and SHA_MR.SMOD must be configured to 2.
  **Note:** When automatic padding is enabled and AES_MR.SMOD = 2 , AES_MR.DUALBUFF must be cleared.
- The SHA_MSR.MSGSIZE must be configured with the length of the authentication message including the optional extended sequence number (ESN) and header and trailer information required by the authentication algorithm used (HMAC, etc.). Refer to the section “Secure Hash Algorithm (SHA)” for more details on configurations for optimized processing of the header information.
- The Security Parameter Index (SPI, sequence number (SEQ#)) and the optional Initialization Vector (IV) must be configured sequentially in SHA_IDATAR0.
- A first DMA transfer descriptor must be configured to transfer the input frame from the system memory to the AES input data registers (AES_IDATARx), and a second DMA descriptor must be configured to transfer the encrypted frame from AES to the system memory.
  **Note:** If AES_EMR.PLIPEN = 1 , there is no need to define a transfer descriptor to load the encrypted frame into the SHA input data registers because the transfer is automatically performed while the second descriptor transfer is in progress.

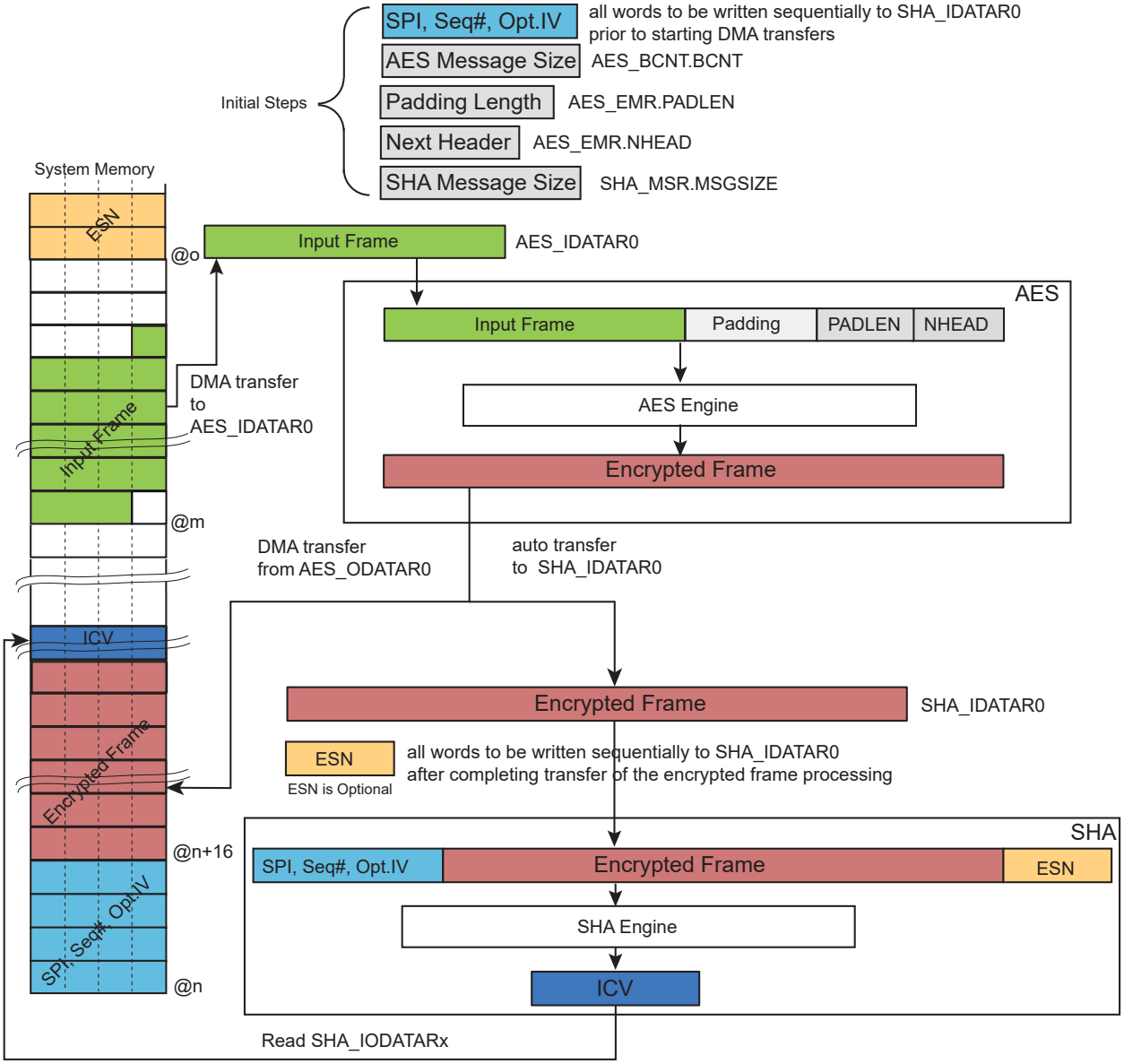
See the figures below.

**Figure 59-12. Generation of an ESP IPSec Frame without ESN**




If the optional extended sequence number is required for authentication, wait for the AES-to-system memory DMA buffer transfer to complete before configuring the ESN value. The ESN value must be configured in the SHA by writing sequentially each 32-bit word of the ESN into the SHA_IDATAR0 register. Wait for SHA_ISR.WRDY=1 before each write in the SHA_IDATAR0 register. See the figure below.

**Figure 59-13. Generation of an ESP IPSec Frame with ESN**




To decipher an ESP IPSec frame without the optional ESN trailer information, two DMA channels are required and the SHA must be configured in Automatic padding mode.

**Note:** AES automatic padding must be disabled when deciphering a frame.

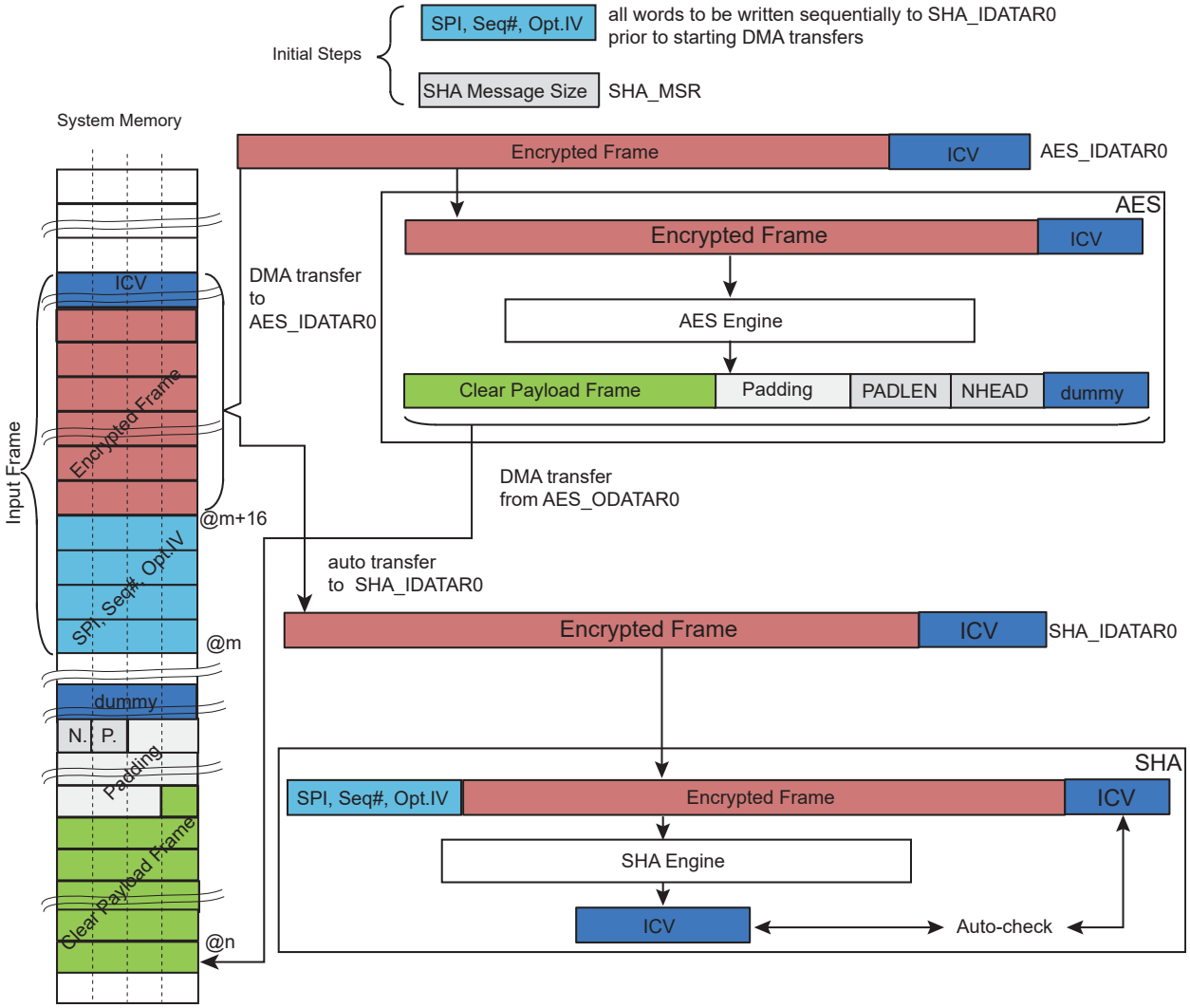
- A first DMA transfer descriptor must be configured to load the received encrypted frame from the system memory to AES_IDATARx for decryption. The start address of the first transfer descriptor must be defined after the SPI, SEQ#, and optional IV (see the figure below).
- A second DMA descriptor must be configured to transfer the decrypted frame from AES_ODATARx to the system memory.
- AES_EMR.PLIPEN and AES_EMR.PLIPD must be written to '1' so that the data buffer is written in AES_IDATARx and in SHA_IDATARx.

The SHA has the capability to perform an automatic check with an expected integrity check value if this value is appended at the end of the frame buffer (SHA_MR.CHECK=2). Thus, if the first transfer

descriptor includes the ICV for SHA, the first DMA transfer allows the decryption and authentication processes including the automatic check. The decrypted part resulting from ICV is not required for downstream processing and must be considered as dummy data.

The end of the decryption and authentication processes occur when flag SHA_ISR.CHECKF=1. The authentication status is provided by SHA_ISR.CHKST.

**Figure 59-14. Decryption of an ESP IP Sec Frame without ESN**

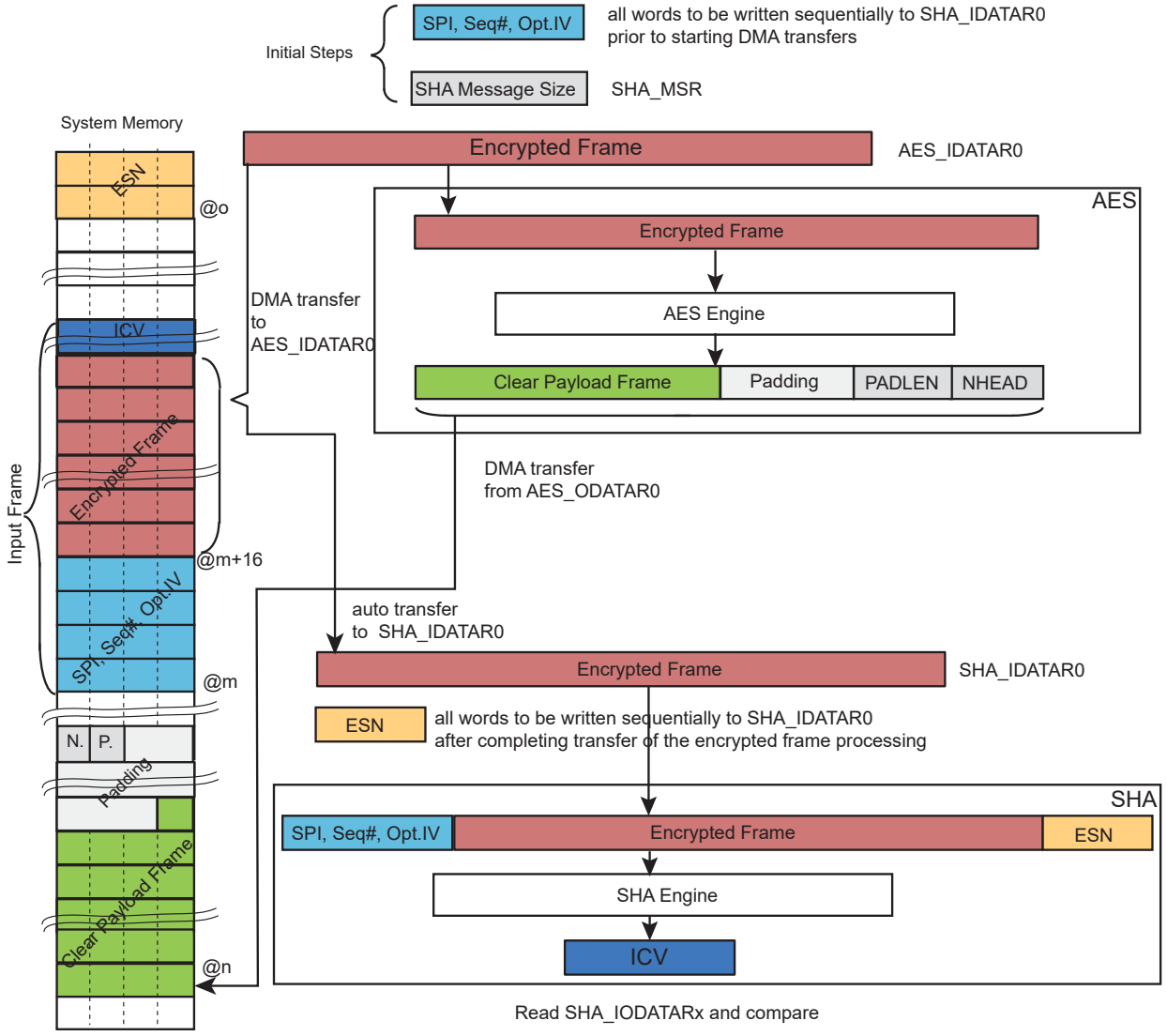



If the optional ESN trailer information is part of the ICV (see the figure below), the ESN must be manually written into SHA_IDATAR0. The ESN value must be written after completion of the system memory-to-AES DMA buffer transfer. The ESN value must be configured in the SHA by writing sequentially each 32-bit word of the ESN into the SHA_IDATAR0 register. Wait for SHA_ISR.WRDY=1 before each write in the SHA_IDATAR0 register.

When the optional ESN trailer information is part of the ICV, it is not possible to include the ICV received in the input frame to the first transfer descriptor. Moreover, if the HMAC algorithm is used for authentication, no automatic check can be performed when optimizing the processing performances of the SHA module. For more details, refer to the section “Secure Hash Algorithm (SHA)”. The result of the HMAC read in the SHA_IODATARx must be manually compared with the ICV value of the input frame.

The comparison must be performed after the end of the authentication process. The authentication process is completed when the SHA_ISR.DATRDY flag is set.

**Figure 59-15. Decryption of an ESP IPSec Frame with ESN**




## 59.4.10 Security Features

### 59.4.10.1 Unspecified Register Access Detection

When an unspecified register access occurs, AES_ISR.URAD is raised. Its source is then reported in AES_ISR.URAT. Only the last unspecified register access is available through the AES_ISR.URAT.

Several kinds of unspecified register accesses can occur:

- Input Data register written during the data processing when SMOD = IDATAR0_START
- Output Data register read during data processing
- Mode register written during data processing
- Output Data register read during sub-keys generation
- Mode register written during sub-keys generation

- Write-only register read access

AES_ISR.URAD and AES_ISR.URAT can only be reset by AES_CR.SWRST.

## 59.5 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | AES_CR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | SWRST |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | AES_MR | 7:0 | PROCDLY[3:0] | | | | DUALBUFF | | GTAGEN | CIPHER |
| | | 15:8 | LOD | OPMOD[2:0] | | | KEYSIZE[1:0] | | SMOD[1:0] | |
| | | 23:16 | CKEY[3:0] | | | | | CFBS[2:0] | | |
| | | 31:24 | | | | | | | | |
| 0x08 ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | AES_IER | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | URAD |
| | | 23:16 | | | | | | PLENERR | EOPAD | TAGRDY |
| | | 31:24 | | | | | | | | |
| 0x14 | AES_IDR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | URAD |
| | | 23:16 | | | | | | PLENERR | EOPAD | TAGRDY |
| | | 31:24 | | | | | | | | |
| 0x18 | AES_IMR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | URAD |
| | | 23:16 | | | | | | PLENERR | EOPAD | TAGRDY |
| | | 31:24 | | | | | | | | |
| 0x1C | AES_ISR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | URAT[3:0] | | | | | | | URAD |
| | | 23:16 | | | | | | PLENERR | EOPAD | TAGRDY |
| | | 31:24 | | | | | | | | |
| 0x20 | AES_KEYWR0 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |
| 0x24 | AES_KEYWR1 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |
| 0x28 | AES_KEYWR2 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |
| 0x2C | AES_KEYWR3 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x30 | AES_KEYWR4 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |
| 0x34 | AES_KEYWR5 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |
| 0x38 | AES_KEYWR6 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |
| 0x3C | AES_KEYWR7 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |
| 0x40 | AES_IDATAR0 | 7:0 | IDATA[7:0] | | | | | | | |
| | | 15:8 | IDATA[15:8] | | | | | | | |
| | | 23:16 | IDATA[23:16] | | | | | | | |
| | | 31:24 | IDATA[31:24] | | | | | | | |
| 0x44 | AES_IDATAR1 | 7:0 | IDATA[7:0] | | | | | | | |
| | | 15:8 | IDATA[15:8] | | | | | | | |
| | | 23:16 | IDATA[23:16] | | | | | | | |
| | | 31:24 | IDATA[31:24] | | | | | | | |
| 0x48 | AES_IDATAR2 | 7:0 | IDATA[7:0] | | | | | | | |
| | | 15:8 | IDATA[15:8] | | | | | | | |
| | | 23:16 | IDATA[23:16] | | | | | | | |
| | | 31:24 | IDATA[31:24] | | | | | | | |
| 0x4C | AES_IDATAR3 | 7:0 | IDATA[7:0] | | | | | | | |
| | | 15:8 | IDATA[15:8] | | | | | | | |
| | | 23:16 | IDATA[23:16] | | | | | | | |
| | | 31:24 | IDATA[31:24] | | | | | | | |
| 0x50 | AES_ODATAR0 | 7:0 | ODATA[7:0] | | | | | | | |
| | | 15:8 | ODATA[15:8] | | | | | | | |
| | | 23:16 | ODATA[23:16] | | | | | | | |
| | | 31:24 | ODATA[31:24] | | | | | | | |
| 0x54 | AES_ODATAR1 | 7:0 | ODATA[7:0] | | | | | | | |
| | | 15:8 | ODATA[15:8] | | | | | | | |
| | | 23:16 | ODATA[23:16] | | | | | | | |
| | | 31:24 | ODATA[31:24] | | | | | | | |
| 0x58 | AES_ODATAR2 | 7:0 | ODATA[7:0] | | | | | | | |
| | | 15:8 | ODATA[15:8] | | | | | | | |
| | | 23:16 | ODATA[23:16] | | | | | | | |
| | | 31:24 | ODATA[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x5C | AES_ODATAR3 | 7:0 | ODATA[7:0] | | | | | | | |
| | | 15:8 | ODATA[15:8] | | | | | | | |
| | | 23:16 | ODATA[23:16] | | | | | | | |
| | | 31:24 | ODATA[31:24] | | | | | | | |
| 0x60 | AES_IVR0 | 7:0 | IV[7:0] | | | | | | | |
| | | 15:8 | IV[15:8] | | | | | | | |
| | | 23:16 | IV[23:16] | | | | | | | |
| | | 31:24 | IV[31:24] | | | | | | | |
| 0x64 | AES_IVR1 | 7:0 | IV[7:0] | | | | | | | |
| | | 15:8 | IV[15:8] | | | | | | | |
| | | 23:16 | IV[23:16] | | | | | | | |
| | | 31:24 | IV[31:24] | | | | | | | |
| 0x68 | AES_IVR2 | 7:0 | IV[7:0] | | | | | | | |
| | | 15:8 | IV[15:8] | | | | | | | |
| | | 23:16 | IV[23:16] | | | | | | | |
| | | 31:24 | IV[31:24] | | | | | | | |
| 0x6C | AES_IVR3 | 7:0 | IV[7:0] | | | | | | | |
| | | 15:8 | IV[15:8] | | | | | | | |
| | | 23:16 | IV[23:16] | | | | | | | |
| | | 31:24 | IV[31:24] | | | | | | | |
| 0x70 | AES_AADLENR | 7:0 | AADLEN[7:0] | | | | | | | |
| | | 15:8 | AADLEN[15:8] | | | | | | | |
| | | 23:16 | AADLEN[23:16] | | | | | | | |
| | | 31:24 | AADLEN[31:24] | | | | | | | |
| 0x74 | AES_CLENR | 7:0 | CLEN[7:0] | | | | | | | |
| | | 15:8 | CLEN[15:8] | | | | | | | |
| | | 23:16 | CLEN[23:16] | | | | | | | |
| | | 31:24 | CLEN[31:24] | | | | | | | |
| 0x78 | AES_GHASHR0 | 7:0 | GHASH[7:0] | | | | | | | |
| | | 15:8 | GHASH[15:8] | | | | | | | |
| | | 23:16 | GHASH[23:16] | | | | | | | |
| | | 31:24 | GHASH[31:24] | | | | | | | |
| 0x7C | AES_GHASHR1 | 7:0 | GHASH[7:0] | | | | | | | |
| | | 15:8 | GHASH[15:8] | | | | | | | |
| | | 23:16 | GHASH[23:16] | | | | | | | |
| | | 31:24 | GHASH[31:24] | | | | | | | |
| 0x80 | AES_GHASHR2 | 7:0 | GHASH[7:0] | | | | | | | |
| | | 15:8 | GHASH[15:8] | | | | | | | |
| | | 23:16 | GHASH[23:16] | | | | | | | |
| | | 31:24 | GHASH[31:24] | | | | | | | |
| 0x84 | AES_GHASHR3 | 7:0 | GHASH[7:0] | | | | | | | |
| | | 15:8 | GHASH[15:8] | | | | | | | |
| | | 23:16 | GHASH[23:16] | | | | | | | |
| | | 31:24 | GHASH[31:24] | | | | | | | |

.....continued

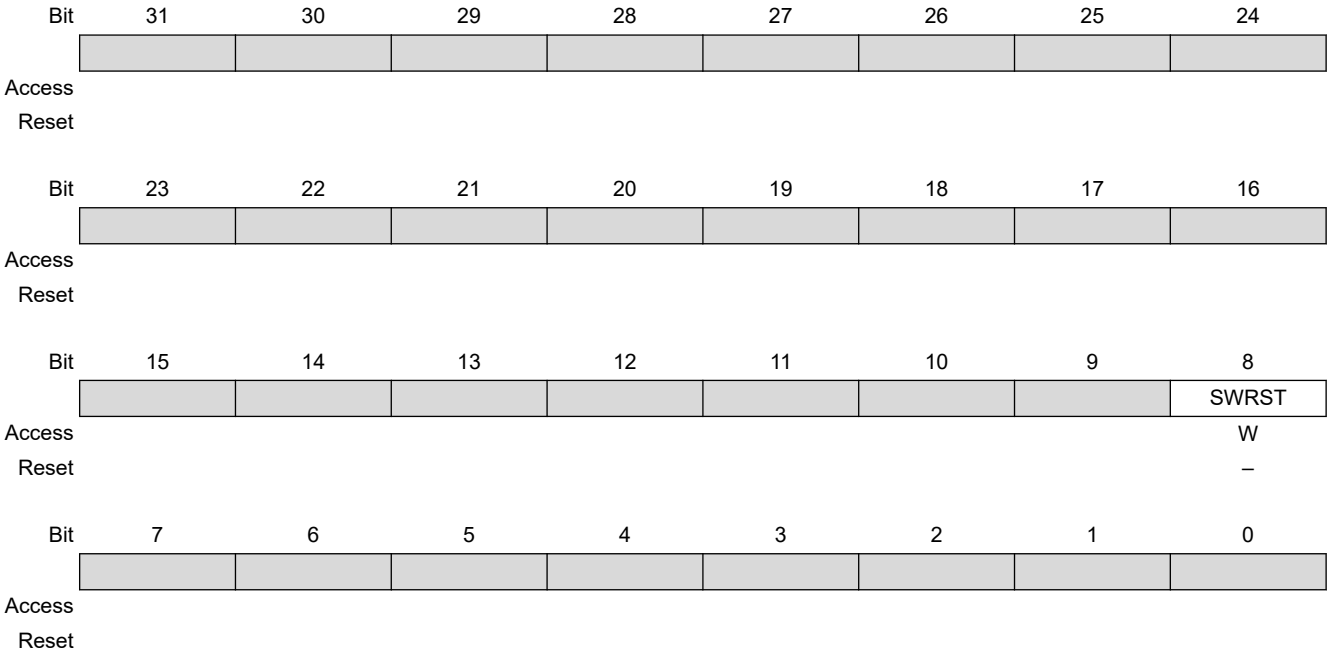
| Offset | Name | Bit Pos. | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x88 | AES_TAGR0 | 7:0 | TAG[7:0] | | | | | | | | |
| | | 15:8 | TAG[15:8] | | | | | | | | |
| | | 23:16 | TAG[23:16] | | | | | | | | |
| | | 31:24 | TAG[31:24] | | | | | | | | |
| 0x8C | AES_TAGR1 | 7:0 | TAG[7:0] | | | | | | | | |
| | | 15:8 | TAG[15:8] | | | | | | | | |
| | | 23:16 | TAG[23:16] | | | | | | | | |
| | | 31:24 | TAG[31:24] | | | | | | | | |
| 0x90 | AES_TAGR2 | 7:0 | TAG[7:0] | | | | | | | | |
| | | 15:8 | TAG[15:8] | | | | | | | | |
| | | 23:16 | TAG[23:16] | | | | | | | | |
| | | 31:24 | TAG[31:24] | | | | | | | | |
| 0x94 | AES_TAGR3 | 7:0 | TAG[7:0] | | | | | | | | |
| | | 15:8 | TAG[15:8] | | | | | | | | |
| | | 23:16 | TAG[23:16] | | | | | | | | |
| | | 31:24 | TAG[31:24] | | | | | | | | |
| 0x98 | AES_CTRR | 7:0 | CTR[7:0] | | | | | | | | |
| | | 15:8 | CTR[15:8] | | | | | | | | |
| | | 23:16 | CTR[23:16] | | | | | | | | |
| | | 31:24 | CTR[31:24] | | | | | | | | |
| 0x9C | AES_GCMHR0 | 7:0 | H[7:0] | | | | | | | | |
| | | 15:8 | H[15:8] | | | | | | | | |
| | | 23:16 | H[23:16] | | | | | | | | |
| | | 31:24 | H[31:24] | | | | | | | | |
| 0xA0 | AES_GCMHR1 | 7:0 | H[7:0] | | | | | | | | |
| | | 15:8 | H[15:8] | | | | | | | | |
| | | 23:16 | H[23:16] | | | | | | | | |
| | | 31:24 | H[31:24] | | | | | | | | |
| 0xA4 | AES_GCMHR2 | 7:0 | H[7:0] | | | | | | | | |
| | | 15:8 | H[15:8] | | | | | | | | |
| | | 23:16 | H[23:16] | | | | | | | | |
| | | 31:24 | H[31:24] | | | | | | | | |
| 0xA8 | AES_GCMHR3 | 7:0 | H[7:0] | | | | | | | | |
| | | 15:8 | H[15:8] | | | | | | | | |
| | | 23:16 | H[23:16] | | | | | | | | |
| | | 31:24 | H[31:24] | | | | | | | | |
| 0xAC ... 0xAF | Reserved | | | | | | | | | | |
| 0xB0 | AES_EMR | 7:0 | | | PLIPD | PLIPEN | | | APM | APEN | |
| | | 15:8 | PADLEN[7:0] | | | | | | | | |
| | | 23:16 | NHEAD[7:0] | | | | | | | | |
| | | 31:24 | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xB4 | AES_BCNT | 7:0 | BCNT[7:0] | | | | | | | |
| | | 15:8 | BCNT[15:8] | | | | | | | |
| | | 23:16 | BCNT[23:16] | | | | | | | |
| | | 31:24 | BCNT[31:24] | | | | | | | |
| 0xB8 ... 0xBF | Reserved | | | | | | | | | |
| 0xC0 | AES_TWR0 | 7:0 | TWEAK[7:0] | | | | | | | |
| | | 15:8 | TWEAK[15:8] | | | | | | | |
| | | 23:16 | TWEAK[23:16] | | | | | | | |
| | | 31:24 | TWEAK[31:24] | | | | | | | |
| 0xC4 | AES_TWR1 | 7:0 | TWEAK[7:0] | | | | | | | |
| | | 15:8 | TWEAK[15:8] | | | | | | | |
| | | 23:16 | TWEAK[23:16] | | | | | | | |
| | | 31:24 | TWEAK[31:24] | | | | | | | |
| 0xC8 | AES_TWR2 | 7:0 | TWEAK[7:0] | | | | | | | |
| | | 15:8 | TWEAK[15:8] | | | | | | | |
| | | 23:16 | TWEAK[23:16] | | | | | | | |
| | | 31:24 | TWEAK[31:24] | | | | | | | |
| 0xCC | AES_TWR3 | 7:0 | TWEAK[7:0] | | | | | | | |
| | | 15:8 | TWEAK[15:8] | | | | | | | |
| | | 23:16 | TWEAK[23:16] | | | | | | | |
| | | 31:24 | TWEAK[31:24] | | | | | | | |
| 0xD0 | AES_ALPHAR0 | 7:0 | ALPHA[7:0] | | | | | | | |
| | | 15:8 | ALPHA[15:8] | | | | | | | |
| | | 23:16 | ALPHA[23:16] | | | | | | | |
| | | 31:24 | ALPHA[31:24] | | | | | | | |
| 0xD4 | AES_ALPHAR1 | 7:0 | ALPHA[7:0] | | | | | | | |
| | | 15:8 | ALPHA[15:8] | | | | | | | |
| | | 23:16 | ALPHA[23:16] | | | | | | | |
| | | 31:24 | ALPHA[31:24] | | | | | | | |
| 0xD8 | AES_ALPHAR2 | 7:0 | ALPHA[7:0] | | | | | | | |
| | | 15:8 | ALPHA[15:8] | | | | | | | |
| | | 23:16 | ALPHA[23:16] | | | | | | | |
| | | 31:24 | ALPHA[31:24] | | | | | | | |
| 0xDC | AES_ALPHAR3 | 7:0 | ALPHA[7:0] | | | | | | | |
| | | 15:8 | ALPHA[15:8] | | | | | | | |
| | | 23:16 | ALPHA[23:16] | | | | | | | |
| | | 31:24 | ALPHA[31:24] | | | | | | | |

### 59.5.1 AES Control Register

**Name:** AES_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only




**Bit 8 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the AES. A software-triggered hardware reset of the AES interface is performed. |

### 59.5.2 AES Mode Register

**Name:** AES_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CKEY[3:0] | | | | | CFBS[2:0] | | |
| Access | W | W | W | W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | – | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LOD | OPMOD[2:0] | | | KEYSIZE[1:0] | | SMOD[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PROCDLY[3:0] | | | | DUALBUFF | | GTAGEN | CIPHER |
| Access | R/W | R/W | R/W | R/W | R/W | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | | 0 | 0 |

**Bits 23:20 – CKEY[3:0]** Key

| Value | Name | Description |
|---|---|---|
| 0xE | PASSWD | This field must be written with 0xE the first time AES_MR is programmed. For subsequent programming of AES_MR, any value can be written, including that of 0xE.<br>Always reads as 0. |

**Bits 18:16 – CFBS[2:0]** Cipher Feedback Data Size

| Value | Name | Description |
|---|---|---|
| 0 | SIZE_128BIT | 128-bit |
| 1 | SIZE_64BIT | 64-bit |
| 2 | SIZE_32BIT | 32-bit |
| 3 | SIZE_16BIT | 16-bit |
| 4 | SIZE_8BIT | 8-bit |

**Bit 15 – LOD** Last Output Data Mode

**WARNING** In DMA mode, reading to the Output Data registers before the last data encryption/decryption process may lead to unpredictable results.

| Value | Description |
|---|---|
| 0 | No effect.<br>After each end of encryption/decryption, the output data are available either on the output data registers (Manual and Auto modes) or at the address specified in the Channel Buffer Transfer Descriptor for DMA mode.<br>In Manual and Auto modes, the DATRDY flag is cleared when at least one of the Output Data registers is read. |
| 1 | The DATRDY flag is cleared when at least one of the Input Data Registers is written.<br>No more Output Data Register reads are necessary between consecutive encryptions/decryptions (see Last Output Data Mode). |

**Bits 14:12 – OPMOD[2:0]** Operating Mode
For CBC-MAC operating mode, set OPMOD to CBC and LOD to 1.

| Value | Name | Description |
|---|---|---|
| 0 | ECB | ECB: Electronic Codebook mode |
| 1 | CBC | CBC: Cipher Block Chaining mode |
| 2 | OFB | OFB: Output Feedback mode |
| 3 | CFB | CFB: Cipher Feedback mode |
| 4 | CTR | CTR: Counter mode (16-bit internal counter) |
| 5 | GCM | GCM: Galois/Counter mode |
| 6 | XTS | XTS: XEX-based tweaked-codebook mode |

**Bits 11:10 – KEYSIZE[1:0]** Key Size

| Value | Name | Description |
|---|---|---|
| 0 | AES128 | AES Key Size is 128 bits |
| 1 | AES192 | AES Key Size is 192 bits |
| 2 | AES256 | AES Key Size is 256 bits |

**Bits 9:8 – SMOD[1:0]** Start Mode
If a DMA transfer is used, configure SMOD to 2. See DMA Mode for more details.

| Value | Name | Description |
|---|---|---|
| 0 | MANUAL_START | Manual Mode |
| 1 | AUTO_START | Auto Mode |
| 2 | IDATAR0_START | AES_IDATAR0 access only Auto Mode (DMA) |

**Bits 7:4 – PROCDLY[3:0]** Processing Delay
Processing Time = N × (PROCDLY + 1)

where

- N = 10 when KEYSIZE = 0

- N = 12 when KEYSIZE = 1
- N = 14 when KEYSIZE = 2

The processing time represents the number of clock cycles that the AES needs in order to perform one encryption/decryption.

**Note:** The best performance is achieved with PROCDLY equal to 0.

**Bit 3 – DUALBUFF** Dual Input Buffer

| Value | Name | Description |
|---|---|---|
| 0 | INACTIVE | AES_IDATARx cannot be written during processing of previous block. |
| 1 | ACTIVE | AES_IDATARx can be written during processing of previous block when SMOD = 2. It speeds up the overall runtime of large files. |

**Bit 1 – GTAGEN** GCM Automatic Tag Generation Enable

| Value | Description |
|---|---|
| 0 | Automatic GCM Tag generation disabled. |
| 1 | Automatic GCM Tag generation enabled. |

**Bit 0 – CIPHER** Processing Mode

| Value | Description |
|---|---|
| 0 | Decrypts data. |
| 1 | Encrypts data. |

### 59.5.3 AES Interrupt Enable Register

**Name:** AES_IER
**Offset:** 0x10
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PLENERR | EOPAD | TAGRDY |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 18 – PLENERR** Padding Length Error Interrupt Enable

**Bit 17 – EOPAD** End of Padding Interrupt Enable

**Bit 16 – TAGRDY** GCM Tag Ready Interrupt Enable

**Bit 8 – URAD** Unspecified Register Access Detection Interrupt Enable

**Bit 0 – DATRDY** Data Ready Interrupt Enable

### 59.5.4 AES Interrupt Disable Register

**Name:** AES_IDR
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PLENERR | EOPAD | TAGRDY |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 18 – PLENERR** Padding Length Error Interrupt Disable

**Bit 17 – EOPAD** End of Padding Interrupt Disable

**Bit 16 – TAGRDY** GCM Tag Ready Interrupt Disable

**Bit 8 – URAD** Unspecified Register Access Detection Interrupt Disable

**Bit 0 – DATRDY** Data Ready Interrupt Disable

### 59.5.5 AES Interrupt Mask Register

**Name:** AES_IMR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PLENERR | EOPAD | TAGRDY |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 18 – PLENERR** Padding Length Error Interrupt Mask

**Bit 17 – EOPAD** End of Padding Interrupt Mask

**Bit 16 – TAGRDY** GCM Tag Ready Interrupt Mask

**Bit 8 – URAD** Unspecified Register Access Detection Interrupt Mask

**Bit 0 – DATRDY** Data Ready Interrupt Mask

### 59.5.6 AES Interrupt Status Register

**Name:** AES_ISR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PLENERR | EOPAD | TAGRDY |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | URAT[3:0] | | | | | | | URAD |
| Access | R | R | R | R | | | | R |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 18 – PLENERR** Padding Length Error

| Value | Description |
|---|---|
| 0 | No Padding Length Error occurred. |
| 1 | Padding Length Error detected. |

**Bit 17 – EOPAD** End of Padding

| Value | Description |
|---|---|
| 0 | Padding is not over. |
| 1 | Padding phase is over. |

**Bit 16 – TAGRDY** GCM Tag Ready

| Value | Description |
|---|---|
| 0 | GCM Tag is not valid. |
| 1 | GCM Tag generation is complete (cleared by reading GCM Tag, starting another processing or when writing a new key). |

**Bits 15:12 – URAT[3:0]** Unspecified Register Access (cleared by writing SWRST in AES_CR)
Only the last Unspecified Register Access Type is available through the URAT field.

| Value | Name | Description |
|---|---|---|
| 0 | IDR_WR_PROCESSING | Input Data register written during the data processing when SMOD = 2 mode. |
| 1 | ODR_RD_PROCESSING | Output Data register read during the data processing. |
| 2 | MR_WR_PROCESSING | Mode register written during the data processing. |
| 3 | ODR_RD_SUBKGEN | Output Data register read during the sub-keys generation. |
| 4 | MR_WR_SUBKGEN | Mode register written during the sub-keys generation. |
| 5 | WOR_RD_ACCESS | Write-only register read access. |

**Bit 8 – URAD** Unspecified Register Access Detection Status (cleared by writing SWRST in AES_CR)

| Value | Description |
|---|---|
| 0 | No unspecified register access has been detected since the last SWRST. |
| 1 | At least one unspecified register access has been detected since the last SWRST. |

**Bit 0 – DATRDY** Data Ready (cleared by setting bit START or bit SWRST in AES_CR or by reading AES_ODATARx)

| Value | Description |
|---|---|
| 0 | Output data not valid. |
| 1 | Encryption or decryption process is completed. |

**Note:** If AES_MR.LOD = 1: In Manual and Auto mode, the DATRDY flag can also be cleared by writing at least one AES_IDATARx.

### 59.5.7 AES Key Word Register x

**Name:** AES_KEYWRx
**Offset:** 0x20 + x*0x04 [x=0..7]
**Reset:** –
**Property:** Write-only

These registers are write-only to prevent the key from being read by another application.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEYW[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEYW[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | KEYW[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | KEYW[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – KEYW[31:0]** Key Word
The four/six/eight 32-bit Key Word registers set the 128-bit/192-bit/256-bit cryptographic key used for AES encryption/decryption.

AES_KEYWR0 corresponds to the first word of the key and respectively AES_KEYWR3/AES_KEYWR5/AES_KEYWR7 to the last one.

Whenever a new key (AES_KEYWRx) is written to the hardware, two automatic actions are processed:

- GCM hash subkey generation
- AES_GHASHRx Clear

See Key Writing and Automatic Hash Subkey Calculation for details.

### 59.5.8 AES Input Data Register x

**Name:** AES_IDATARx
**Offset:** 0x40 + x*0x04 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – IDATA[31:0]** Input Data Word
The four 32-bit Input Data registers set the 128-bit data block used for encryption/decryption.

AES_IDATAR0 corresponds to the first word of the data to be encrypted/decrypted, and AES_IDATAR3 to the last one.

These registers are write-only to prevent the input data from being read by another application.

### 59.5.9 AES Output Data Register x

**Name:** AES_ODATARx
**Offset:** 0x50 + x*0x04 [x=0..3]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ODATA[31:0]** Output Data
The four 32-bit Output Data registers contain the 128-bit data block that has been encrypted/decrypted.

AES_ODATAR0 corresponds to the first word, AES_ODATAR3 to the last one.

### 59.5.10 AES Initialization Vector Register x

**Name:** AES_IVRx
**Offset:** 0x60 + x*0x04 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IV[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IV[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IV[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IV[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – IV[31:0]** Initialization Vector
The four 32-bit Initialization Vector registers set the 128-bit Initialization Vector data block that is used by some modes of operation as an additional initial input.

AES_IVR0 corresponds to the first word of the Initialization Vector, AES_IVR3 to the last one.

These registers are write-only to prevent the Initialization Vector from being read by another application.

For CBC, OFB and CFB modes, the IV input value corresponds to the initialization vector.

For CTR mode, the IV input value corresponds to the initial counter value.

**Note:** These registers are not used in ECB mode and must not be written.

### 59.5.11 AES Additional Authenticated Data Length Register

**Name:** AES_AADLENR
**Offset:** 0x70
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | AADLEN[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | AADLEN[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | AADLEN[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | AADLEN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – AADLEN[31:0]** Additional Authenticated Data Length
Length in bytes of the Additional Authenticated Data (AAD) that is to be processed.

**Note:** The maximum byte length of the AAD portion of a message is limited to the 32-bit counter length.

### 59.5.12 AES Plaintext/Ciphertext Length Register

**Name:** AES_CLENR
**Offset:** 0x74
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CLEN[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CLEN[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CLEN[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CLEN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CLEN[31:0]** Plaintext/Ciphertext Length
Length in bytes of the plaintext/ciphertext (C) data that is to be processed.

**Note:** The maximum byte length of the C portion of a message is limited to the 32-bit counter length.

### 59.5.13 AES GCM Intermediate Hash Word Register x

**Name:** AES_GHASHRx
**Offset:** 0x78 + x*0x04 [x=0..3]
**Reset:** 0x00000000
**Property:** R/W

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | GHASH[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GHASH[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GHASH[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | GHASH[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – GHASH[31:0]** Intermediate GCM Hash Word x
The four 32-bit Intermediate Hash Word registers expose the intermediate GHASH value. May be read to save the current GHASH value so processing can later be resumed, presumably on a later message fragment. Whenever a new key is written to the AES Key Register two automatic actions are processed:

- GCM hash subkey generation
- AES_GHASHRx Clear

See Key Writing and Automatic Hash Subkey Calculation for details.

If an application software-specific hash initial value is needed for the GHASH, it must be written to AES_GHASHRx:

- after a write to the AES Key Register, if any,
- before starting the input data feed.

### 59.5.14 AES GCM Authentication Tag Word Register x

**Name:** AES_TAGRx
**Offset:** 0x88 + x*0x04 [x=0..3]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TAG[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TAG[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TAG[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TAG[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – TAG[31:0]** GCM Authentication Tag x
The four 32-bit Tag registers contain the final 128-bit GCM Authentication tag (*T*) when GCM processing is complete. TAG0 corresponds to the first word, TAG3 to the last word.

### 59.5.15 AES GCM Encryption Counter Value Register

**Name:** AES_CTRR
**Offset:** 0x98
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CTR[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CTR[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CTR[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CTR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CTR[31:0]** GCM Encryption Counter
Reports the current value of the 32-bit GCM counter.

### 59.5.16 AES GCM H Word Register x

**Name:** AES_GCMHRx
**Offset:** 0x9C + x*0x04 [x=0..3]
**Reset:** 0x00000000
**Property:** R/W

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | H[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | H[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | H[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | H[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – H[31:0]** GCM H Word x
The four 32-bit H Word registers contain the 128-bit GCM hash subkey *H*value.

Whenever a new key is written to the AES Key Register, two automatic actions are processed:

- GCM hash subkey *H* generation
- AES_GHASHRx Clear

If the application software requires a specific hash subkey, the automatically-generated H value can be overwritten in AES_GCMHRx. See Key Writing and Automatic Hash Subkey Calculation for details.

Generating a GCM hash subkey *H* by a write in AES_GCMHRx enables to:

- select the GCM hash subkey *H* for GHASH operations,
- select one operand to process a single GF128 multiply.

### 59.5.17 AES Extended Mode Register

**Name:** AES_EMR
**Offset:** 0xB0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NHEAD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PADLEN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | PLIPD | PLIPEN | | | APM | APEN |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bits 23:16 – NHEAD[7:0]** IPSEC Next Header

| Value | Description |
|---|---|
| 0–255 | IPSEC Next Header field |

**Bits 15:8 – PADLEN[7:0]** Auto Padding Length

| Value | Description |
|---|---|
| 0–255 | Padding length in bytes |

**Bit 5 – PLIPD** Protocol Layer Improved Performance Decipher

| Value | Description |
|---|---|
| 0 | Protocol layer improved performance is in ciphering mode. |
| 1 | Protocol layer improved performance is in deciphering mode. |

**Bit 4 – PLIPEN** Protocol Layer Improved Performance Enable

| Value | Description |
|---|---|
| 0 | Protocol layer improved performance is disabled. |
| 1 | Protocol layer improved performance is enabled. |

**Bit 1 – APM** Auto Padding Mode

| Value | Description |
|---|---|
| 0 | Auto Padding performed according to IPSEC standard. |
| 1 | Auto Padding performed according to SSL standard. |

**Bit 0 – APEN** Auto Padding Enable

| Value | Description |
|---|---|
| 0 | Auto Padding feature is disabled. |
| 1 | Auto Padding feature is enabled. |

### 59.5.18 AES Byte Counter Register

**Name:** AES_BCNT
**Offset:** 0xB4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BCNT[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BCNT[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BCNT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BCNT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – BCNT[31:0]** Auto Padding Byte Counter
Auto padding byte counter value. BCNT must be greater than 0.

### 59.5.19 AES Tweak Word Register x

**Name:** AES_TWRx
**Offset:** 0xC0 + x*0x04 [x=0..3]
**Reset:** 0x00000000
**Property:** R/W

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TWEAK[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TWEAK[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TWEAK[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TWEAK[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – TWEAK[31:0]** Tweak Word x
The four 32-bit Tweak Word registers contain the 128-bit Tweak value.

### 59.5.20 AES Alpha Word Register x

**Name:** AES_ALPHARx
**Offset:** 0xD0 + x*0x04 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ALPHA[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ALPHA[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ALPHA[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ALPHA[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – ALPHA[31:0]** Alpha Word x
The four 32-bit Alpha Word registers contain the 128-bit primitive of GF($2^{128}$) to use for the first processing.

# 60. Secure Hash Algorithm (SHA)

## 60.1 Description

The Secure Hash Algorithm (SHA) is compliant with the American FIPS (Federal Information Processing Standard) Publication 180-2 specification.

The 512/1024-bit block of message is respectively stored in 16/32 x 32-bit registers, (SHA_IDATARx/ SHA_IODATARx) which are write-only.

As soon as the input data is written, the hash processing may be started. The registers comprising the block of a padded message must be entered consecutively. Then the message digest is ready to be read out on the 5 up to 8/16 x 32-bit output data registers (SHA_IODATARx) or through the DMA channels.

## 60.2 Embedded Characteristics

- Supports Hash-based Message Authentication Code (HMAC) Algorithm (HMAC-SHA1, HMAC-SHA224, HMAC-SHA256, HMAC-SHA384, HMAC-SHA512, )
- Compliant with FIPS Publication 180-2
- Supports Automatic Padding of Messages
- Supports Up to 2 Sets of Initial Hash Values Registers (HMAC Acceleration or other)
- Supports Automatic Check of the Hash (HMAC Acceleration or other)
- Tightly Coupled to AES for Protocol Layers Improved Performances
- Configurable Processing Period:
  - 85 Clock Cycles to obtain a fast SHA1 runtime, 88 clock cycles for SHA384, SHA512 or 209 Clock Cycles for Maximizing Bandwidth of Other Applications
  - 72 Clock Cycles to obtain a fast SHA224, SHA256 runtime or 194 Clock Cycles for Maximizing Bandwidth of Other Applications
- Connection to DMA Channel Capabilities Optimizes Data Transfers
- Double Input Buffer Optimizes Runtime

## 60.3 Product Dependencies

### 60.3.1 Power Management

The SHA may be clocked through the Power Management Controller (PMC), so the programmer must first configure the PMC to enable the SHA clock.

### 60.3.2 Interrupt Sources

The SHA interface has an interrupt line connected to the Interrupt Controller.

Handling the SHA interrupt requires programming the Interrupt Controller before configuring the SHA.

## 60.4 Functional Description

The Secure Hash Algorithm (SHA) module requires a padded message according to FIPS180-2 specification. This message can be provided with the padding to the SHA module, or the padding can be

automatically computed by the SHA module if the size of the message is provided. The first block of the message must be indicated to the module by a specific command. The SHA module produces an N-bit message digest each time a block is written and processing period ends, where N is 160 for SHA1, 224 for SHA224, 256 for SHA256, 384 for SHA384, 512 for SHA512. The SHA module is also capable of computing Hash-based Message Authentication Code (HMAC) algorithm.

### 60.4.1 SHA Algorithm

The SHA can process SHA1, SHA224, SHA256, SHA384, SHA512 by configuring the ALGO field in the SHA Mode register (SHA_MR).

### 60.4.2 HMAC Algorithm

The HMAC algorithm is as follows:

$HMAC_K(m) = h((K_0 \oplus opad) || h((K_0 \oplus ipad) || m))$

where:

- h = SHA function
- $K_0$ = the key K after any necessary pre-processing to form a block size key
- m = message to authenticate
- || = concatenation operator
- $\oplus$ = XOR operator
- ipad = predefined constant (0x3636...3636)
- opad = predefined constant (0x5C5C...5C5C)

The SHA provides a fully optimized processing of the HMAC algorithm by executing the following operations:

- starting the SHA algorithm from any user predefined hash value, thus ‘$h(K_0 \oplus ipad)$’ for first HMAC hash and ‘$h(K_0 \oplus opad)$’ for second HMAC hash
- performing automatic padding
- routing automatically the first hash result ‘$h((K_0 \oplus ipad) || m)$’ to the source of the second hash processing ‘$h((K_0 \oplus opad) || (first\ hash\ result))$’ including the concatenation of the first hash result to ‘$K_0 \oplus opad$’.

To perform the HMAC operation, the ALGO field value must be greater than 7, the automatic padding feature must be enabled (MSGSIZE and BYTCNT fields differ from 0) and the SHA internal initial hash value registers 0 and 1 must be configured, respectively, with the hash results of input blocks “$K_0 \oplus ipad$” and “$K_0 \oplus opad$” (see Internal Registers for Initial Hash Value or Expected Hash Result).

The size of the message (‘m’) must be written in the MSGSIZE and BYTCNT fields.

The FIRST bit in the SHA Control register (SHA_CR) should be set before writing the first block of the message.

The SHA can process HMAC-SHA1, HMAC-SHA256, HMAC-SHA384, HMAC-SHA512 by configuring the ALGO field in the SHA_MR.

### 60.4.3 Processing Period

The processing period can be configured.

The short processing period allocates bandwidth to the SHA module, whereas the long processing period allocates more bandwidth on the system bus to other applications. An example is DMA channels not associated with SHA.

In SHA1 mode, the shortest processing period is 85 clock cycles + 2 clock cycles for start command synchronization. The longest period is 209 clock cycles + 2 clock cycles.

In SHA384, SHA512 mode, the shortest processing period is 88 clock cycles + 2 clock cycles for start command synchronization. The longest period is 209 clock cycles + 2 clock cycles.

In SHA256 and SHA224 mode, the shortest processing period is 72 clock cycles + 2 clock cycles for start command synchronization. The longest period is 194 clock cycles + 2 clock cycles.

### 60.4.4 Double Input Buffer

The SHA Input Data registers (SHA_IDATARx) can be double-buffered to reduce the runtime of large files.

Double-buffering allows a new message block to be written while the previous message block is being processed. This is only possible when DMA accesses are performed (SMOD = 2).

The DUALBUFF bit in the SHA_MR must be set to have double input buffer access.

### 60.4.5 Internal Registers for Initial Hash Value or Expected Hash Result

The SHA module embeds two sets of internal registers (IR0, IR1) to store different data used by the SHA or HMAC algorithms (See the figure below). These internal registers are accessed through SHA Input Data registers (SHA_IDATARx).

When the ALGO field selects SHA algorithms, IR0 can be configured with a user initial hash value. This initial hash value can be used to compute a custom hash algorithm with two sets of different initial constants, or to continue a hash computation by providing the intermediate hash value previously returned by the SHA module.

When the ALGO field selects SHA algorithms, IR1 can be configured with either a user initial hash value or an expected hash result. The expected hash result must be configured in the IR1 if the field CHECK = 1 (refer to Automatic Check). If the field CHECK = 0 or 2, IR1 can be configured with a user initial hash value that differs from IR0 value.

When the ALGO field selects HMAC algorithms, IR0 must be configured with the hash result of $K_0 \oplus$ ipad and IR1 must be configured with the hash result of $K_0 \oplus$ opad. These pre-computed first blocks speed up the HMAC computation by saving the time to compute the intermediate hash values of the first block which is constant while the secret key is constant (See HMAC Algorithm).

**Table 60-1. Configuration Values of Internal Registers**

| Register | SHA Modes (ALGO < 8) | | | HMAC Modes (ALGO > 7) |
|---|---|---|---|---|
| | CHECK = 0 | CHECK = 1 | CHECK = 2 | |
| IR0 | User Initial Hash | User Initial Hash | User Initial Hash | hash($K_0 \oplus$ ipad) |
| IR1 | User Initial Hash | Expected Hash Result | User Initial Hash | hash($K_0 \oplus$ opad) |

To calculate the initial HMAC values, follow this sequence:

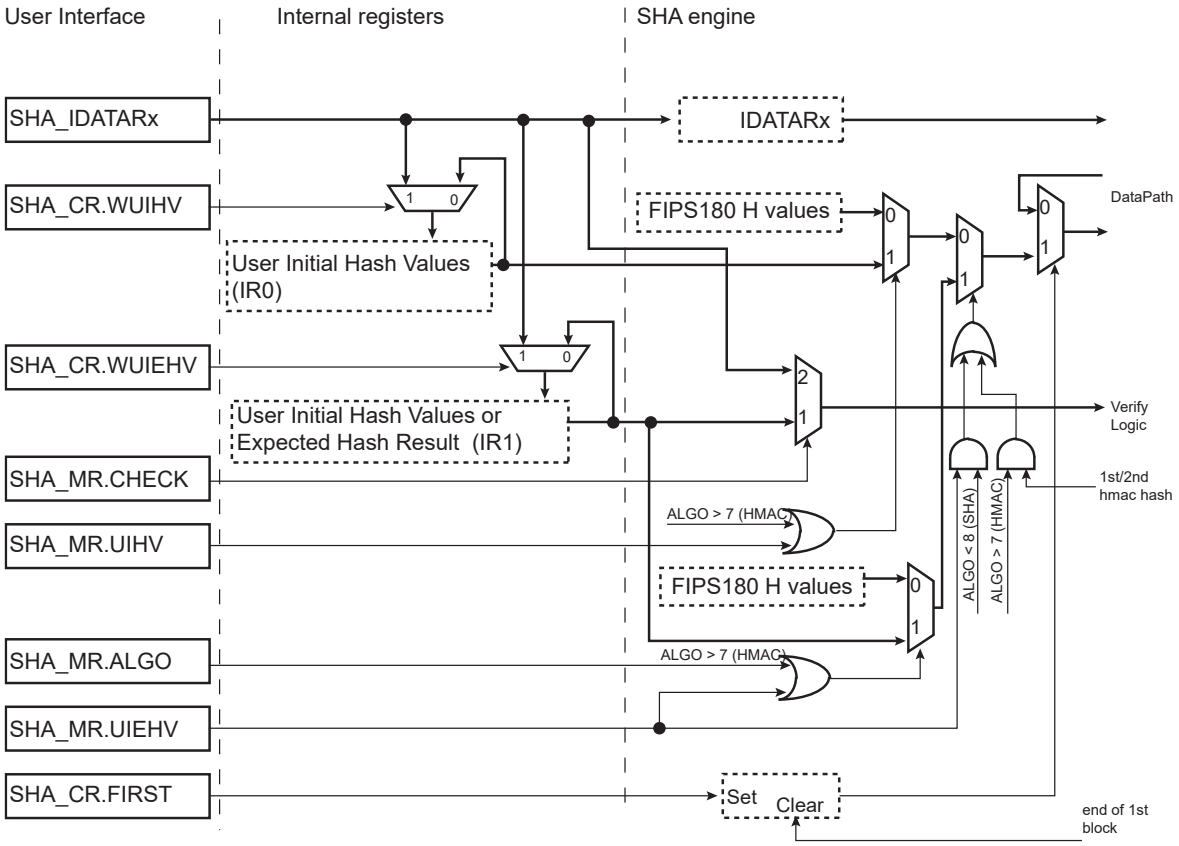
1. Calculate $K_0$.
2. Calculate $K_0 \oplus$ ipad and $K_0 \oplus$ opad.
3. Perform a hash of the result of $K_0 \oplus$ ipad and $K_0 \oplus$ opad (auto-padding must be disabled for that type of hash).
4. Write h($K_0 \oplus$ ipad) and h($K_0 \oplus$ opad) in IR0 and IR1 respectively.

To write IR0 or IR1, follow this sequence:

1. Set SHA_CR. WUIHV (IR0) or SHA_CR.WUIEHV (IR1).
2. Write the data in SHA_IDATARx. The number of registers to write depends on the type of data (user initial hash values or expected hash result) and on the type of algorithm selected:
   – SHA_IDATAR0 to SHA_IDATAR4 for data used in algorithms based on SHA1
   – SHA_IDATAR0 to SHA_IDATAR7 for data used in algorithms based on SHA256
   – SHA_IDATAR0 to SHA_IDATAR15 for data used in algorithms based on SHA512
   – SHA_IDATAR0 to SHA_IDATAR6 for expected hash result of algorithms based on SHA224
   – SHA_IDATAR0 to SHA_IDATAR11 for expected hash result of algorithms based on SHA384
3. Clear SHA_CR.WUIHV or SHA_CR.WUIEHV.

IR0 and IR1 are automatically selected for HMAC processing if the field ALGO selects HMAC algorithms. If SHA algorithms are selected, the internal registers are selected if the corresponding UIHV or UIEHV bits are set.

**Figure 60-1. User Initial Hash Value and Expected Hash Internal Register Access**




### 60.4.6 Automatic Padding

The SHA module features an automatic padding computation to speed up the execution of the algorithm.

The automatic padding function requires the following information:

- Complete message size in bytes to be written in the MSGSIZE field of the SHA Message Size register (SHA_MSR).
  The size of the message is written at the end of the last block, as required by the FIPS180-2 specification (the size is automatically converted into a bit-size).

- Number of remaining bytes (to write in the SHA_IDATARx) to be written in the BYTCNT field of the SHA Bytes Count register (SHA_BCR).
  Automatic padding occurs when the BYTCNT field reaches 0. At each write in the SHA Input registers, the BYTCNT field value is decreased by the number of bytes written.

The BYTCNT field value must be written with the same value as the MSGSIZE field value if the full message is processed. If the message is partially preprocessed and an initial hash value is used, BYTCNT must be written with the remaining bytes to hash while MSGSIZE holds the message size.

To disable the automatic padding feature, the MSGSIZE and BYTCNT fields must be configured with 0.

### 60.4.7 Automatic Check

The SHA module features an automatic check of the hash result with the expected hash. A check failure can generate an interrupt if configured in the SHA Interrupt Enable register (SHA_IER).

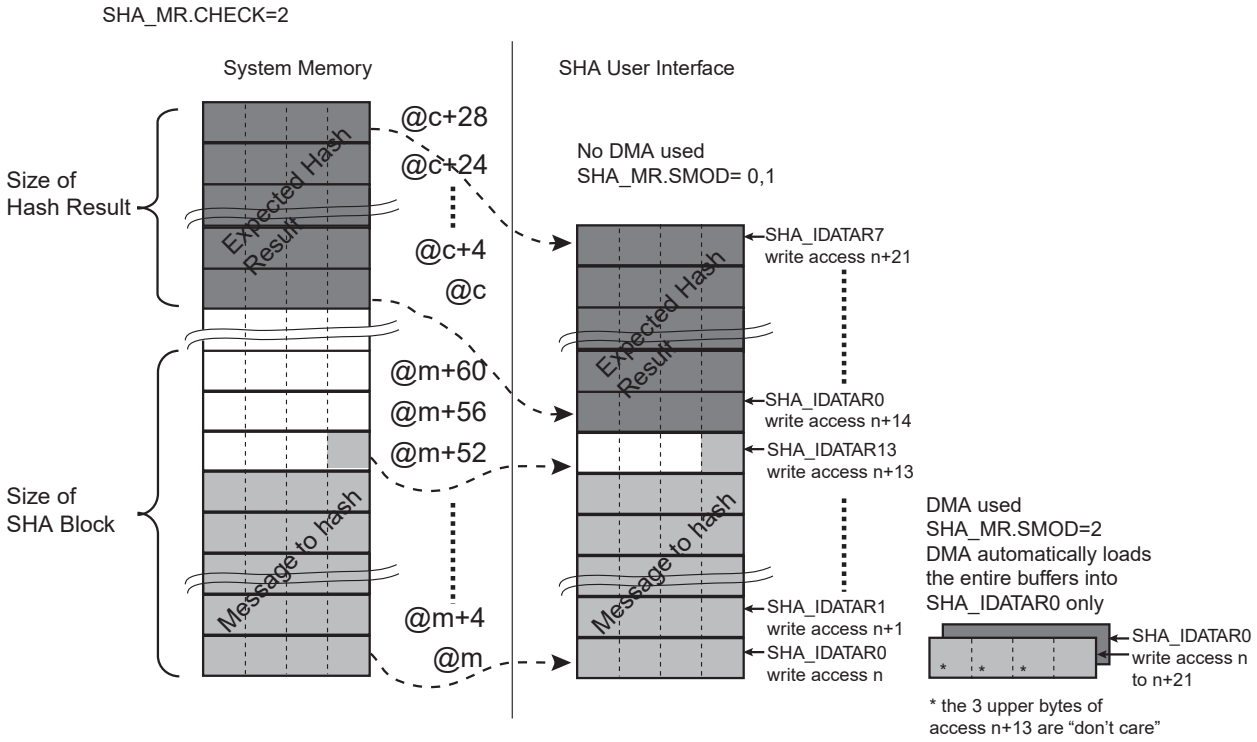
Automatic check requires the automatic padding feature to be enabled (MSGSIZE and BYTCNT fields must be greater than 0).

There are two methods to configure the expected hash result:

- if SHA_MR.CHECK = 1, the expected hash result is read from the internal register (IR1). This method cannot be used when HMAC algorithms is selected because this register is already used to store user initial hash values for the second hash processing. IR1 cannot be read by software.
- If SHA_MR.CHECK = 2, the expected hash result is written in the SHA_IDATARx after the message.

When SHA_MR.CHECK = 2, the method can provide more flexibility of use if a message is stored in system memory together with its expected hash result. A DMA with linked list can be used to ease the transfer of the message and its expected hash result.

**Figure 60-2. Message and Expected Hash Result Memory Mapping**



The number of 32-bit words of the hash result to check with the expected hash can be selected with SHA_MR.CHKCNT. The status of the check is available in the CHKST field in the SHA Interrupt Status register (SHA_ISR).

An interrupt can be generated (if enabled) when the check is completed. The check occurs several clock cycles after the computation of the requested hash, so the interrupt and the CHECKF bit are set several clock cycles after the DATRDY flag of the SHA_ISR.

### 60.4.8 Protocol Layers Improved Performances

The SHA can be tightly coupled to the AES module to improve performances when processing protocol layers such as IPsec or OpenSSL.

When the AES is configured to be tightly coupled to SHA (AES_MR), SHA must be always configured in Double Buffer mode (SHA_MR.DUALBUFF = 1).

Refer to the section “Advanced Encryption Standard (AES)” for details.

### 60.4.9 Start Modes

SHA_MR.SMOD is used to select the Hash Processing Start mode.

#### 60.4.9.1 Manual Mode

In Manual mode, the sequence is as follows:

1. Set SHA_IER.DATRDY (Data Ready) , depending on whether an interrupt is required at the end of processing.
2. If the initial hash values differ from the FIPS standard, set SHA_MR.UIHV and/or SHA_MR.UIEHV. If the initial hash values comply with the FIPS180-2 specification, clear SHA_MR.UIHV and/or SHA_MR.UIEHV.
3. If automatic padding is required, configure SHA_MSR.MSGSIZE with the number of bytes of the message, and configure SHA_BCR.BYTCNT with the remaining number of bytes to write. The BYTCNT field must be written with a value different from MSGSIZE field value if the message is preprocessed and completed by using user initial hash values.
   If automatic padding is not required, configure SHA_MSR.MSGSIZE and SHA_BCR.BYTCNT to 0.
4. For the first block of a message, the FIRST command must be set by writing a 1 into the corresponding bit of the Control register (SHA_CR). For the other blocks, there is nothing to write.
5. Write the block to be processed in the SHA_IDATARx.
6. To begin processing, set SHA_CR.START.
7. When processing is completed, the bit DATRDY in the Interrupt Status register (SHA_ISR) rises. If an interrupt has been enabled by setting SHA_IER.DATRDY, the interrupt line of the SHA is activated.
8. Repeat the write procedure for each block, start procedure and wait for the interrupt procedure up to the last block of the entire message. Each time the start procedure is complete, the DATRDY flag is cleared.
9. After the last block is processed (DATRDY flag is set, if an interrupt has been enabled by setting SHA_IER.DATRDY, the interrupt line of the SHA is activated), read the message digest in the Output Data registers. The DATRDY flag is automatically cleared when reading the SHA_IODATARx registers.

#### 60.4.9.2 Auto Mode

In Auto mode, processing starts as soon as the correct number of SHA_IDATARx is written. No action in the SHA_CR is necessary.

#### 60.4.9.3 DMA Mode

The DMA can be used in association with the SHA to perform the algorithm on a complete message without any action by the software during processing.

SHA_MR.SMOD must be configured to 2.

The DMA must be configured with non-incremental addresses.

The start address of any transfer descriptor must be set to point to the SHA_IDATAR0.

The DMA chunk size must be set to transfer, for each trigger request, 16 words of 32 bits.

The FIRST bit of the SHA_CR must be set before starting the DMA when the first block is transferred.

The DMA generates an interrupt when the end of buffer transfer is completed but the SHA processing is still in progress. The end of SHA processing is indicated by the flag DATRDY in the SHA_ISR.

If automatic padding is disabled, the end of SHA processing requires two interrupts to be verified. The DMA end of transfer interrupt must be verified first, then the SHA DATRDY interrupt must be enabled and verified.

If automatic padding is enabled, the end of SHA processing requires only one interrupt to be verified. The DMA end of transfer is not required, so the SHA DATRDY interrupt must be enabled prior to start the DMA and DATRDY interrupt is the only one to be verified. Refer to the figures below.

**Figure 60-3. Interrupts Processing with DMA**

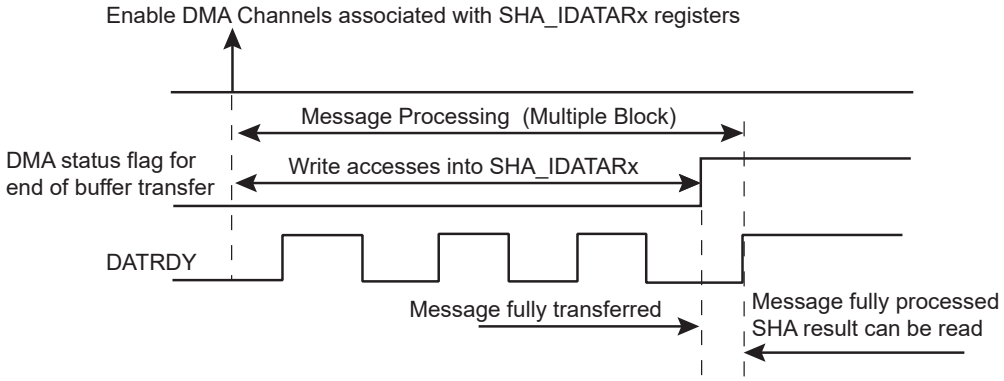
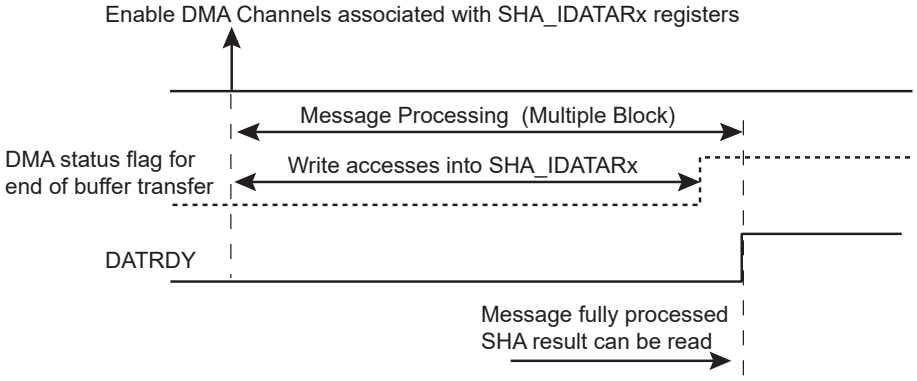



**Figure 60-4. Interrupts Processing with DMA and Automatic Padding**




#### 60.4.9.4 SHA Register Endianness

In ARM processor-based products, the system bus and processors manipulate data in little-endian form. The SHA interface requires little-endian format words. However, in accordance with the protocol of FIPS 180-2 specification, data is collected, processed and stored by the SHA algorithm in big-endian form.

The following example illustrates how to configure the SHA:

If the first 64 bits of a message (according to FIPS 180-2, i.e., big-endian format) to be processed is 0xcafedeca_01234567, then the SHA_IDATAR0 and SHA_IDATAR1 registers must be written with the following pattern:

- SHA_IDATAR0 = 0xcadefeca
- SHA_IDATAR1 = 0x67452301

In a little-endian system, the message (according to FIPS 180-2) starting with pattern 0xcafedeca_01234567 is stored into memory as follows:

- 0xca stored at initial offset (for example 0x00),
- then 0xfe stored at initial offset + 1 (i.e., 0x01),
- 0xde stored at initial offset + 2 (i.e., 0x02),
- 0xca stored at initial offset + 3 (i.e., 0x03).

If the message is received through a serial-to-parallel communication channel, the first received character is 0xca and it is stored at the first memory location (initial offset). The second byte, 0xfe, is stored at initial offset + 1.

When reading on a 32-bit little-endian system bus, the first word read back from system memory is 0xcadefeca.

When the SHA_IODATARx registers are read, the hash result is organized in little-endian format, allowing system memory storage in the same format as the message.

Taking an example from the FIPS 180-2 specification Appendix B.1, the endian conversion can be observed.

For this example, the 512-bit message is:

0x6162638000000000

### 60.4.10 Security Features

#### 60.4.10.1 Unspecified Register Access Detection

When an unspecified register access occurs, the URAD bit in the SHA_ISR is set. Its source is then reported in the Unspecified Register Access Type field (URAT). Only the last unspecified register access is available through the URAT field.

Several kinds of unspecified register accesses can occur:

- SHA_IDATARx written during data processing in DMA mode
- SHA_IODATARx read during data processing
- SHA_MR written during data processing
- Write-only register read access

The URAD bit and the URAT field can only be reset by the SWRST bit in the SHA_CR.

## 60.5 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | SHA_CR | 7:0 | | | | FIRST | | | | START |
| | | 15:8 | | | WUIEHV | WUIHV | | | | SWRST |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | SHA_MR | 7:0 | | UIEHV | UIHV | PROCDLY | | | SMOD[1:0] | |
| | | 15:8 | | | | | ALGO[3:0] | | | |
| | | 23:16 | | | | | | | | DUALBUFF |
| | | 31:24 | CHKCNT[3:0] | | | | | | CHECK[1:0] | |
| 0x08 ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | SHA_IER | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | URAD |
| | | 23:16 | | | | | | | | CHECKF |
| | | 31:24 | | | | | | | | |
| 0x14 | SHA_IDR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | URAD |
| | | 23:16 | | | | | | | | CHECKF |
| | | 31:24 | | | | | | | | |
| 0x18 | SHA_IMR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | URAD |
| | | 23:16 | | | | | | | | CHECKF |
| | | 31:24 | | | | | | | | |
| 0x1C | SHA_ISR | 7:0 | | | | WRDY | | | | DATRDY |
| | | 15:8 | | URAT[2:0] | | | | | | URAD |
| | | 23:16 | CHKST[3:0] | | | | | | | CHECKF |
| | | 31:24 | | | | | | | | |
| 0x20 | SHA_MSR | 7:0 | MSGSIZE[7:0] | | | | | | | |
| | | 15:8 | MSGSIZE[15:8] | | | | | | | |
| | | 23:16 | MSGSIZE[23:16] | | | | | | | |
| | | 31:24 | MSGSIZE[31:24] | | | | | | | |
| 0x24 ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | SHA_BCR | 7:0 | BYTCNT[7:0] | | | | | | | |
| | | 15:8 | BYTCNT[15:8] | | | | | | | |
| | | 23:16 | BYTCNT[23:16] | | | | | | | |
| | | 31:24 | BYTCNT[31:24] | | | | | | | |
| 0x34 ... 0x3F | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x40 | SHA_IDATAR0 | 7:0 | IDATA[7:0] | | | | | | | |
| | | 15:8 | IDATA[15:8] | | | | | | | |
| | | 23:16 | IDATA[23:16] | | | | | | | |
| | | 31:24 | IDATA[31:24] | | | | | | | |
| ... | | | | | | | | | | |
| 0x7C | SHA_IDATAR15 | 7:0 | IDATA[7:0] | | | | | | | |
| | | 15:8 | IDATA[15:8] | | | | | | | |
| | | 23:16 | IDATA[23:16] | | | | | | | |
| | | 31:24 | IDATA[31:24] | | | | | | | |
| 0x80 | SHA_IODATAR0 | 7:0 | IODATA[7:0] | | | | | | | |
| | | 15:8 | IODATA[15:8] | | | | | | | |
| | | 23:16 | IODATA[23:16] | | | | | | | |
| | | 31:24 | IODATA[31:24] | | | | | | | |
| ... | | | | | | | | | | |
| 0xBC | SHA_IODATAR15 | 7:0 | IODATA[7:0] | | | | | | | |
| | | 15:8 | IODATA[15:8] | | | | | | | |
| | | 23:16 | IODATA[23:16] | | | | | | | |
| | | 31:24 | IODATA[31:24] | | | | | | | |

### 60.5.1 SHA Control Register

**Name:** SHA_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | WUIEHV | WUIHV | | | | SWRST |
| Access | | | W | W | | | | W |
| Reset | | | – | – | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | FIRST | | | | START |
| Access | | | | W | | | | W |
| Reset | | | | – | | | | – |

**Bit 13 – WUIEHV** Write User Initial or Expected Hash Values

| Value | Description |
|---|---|
| 0 | SHA_IDATARx accesses are routed to the data registers. |
| 1 | SHA_IDATARx accesses are routed to the internal registers (IR1). |

**Bit 12 – WUIHV** Write User Initial Hash Values

| Value | Description |
|---|---|
| 0 | SHA_IDATARx accesses are routed to the data registers. |
| 1 | SHA_IDATARx accesses are routed to the internal registers (IR0). |

**Bit 8 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the SHA. A software-triggered hardware reset of the SHA interface is performed. |

**Bit 4 – FIRST** First Block of a Message

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Indicates that the next block to process is the first one of a message. |

**Bit 0 – START** Start Processing

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Starts manual hash algorithm process. |

### 60.5.2 SHA Mode Register

**Name:** SHA_MR
**Offset:** 0x04
**Reset:** 0x0000100
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CHKCNT[3:0] | | | | | | CHECK[1:0] | |
| Access | R/W | R/W | R/W | R/W | | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DUALBUFF |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | ALGO[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | UIEHV | UIHV | PROCDLY | | | SMOD[1:0] | |
| Access | | R/W | R/W | R/W | | | R/W | R/W |
| Reset | | 0 | 0 | 0 | | | 0 | 0 |

**Bits 31:28 – CHKCNT[3:0]** Check Counter
Number of 32-bit words to check. The value 0 indicates that the number of words to compare will be based on the algorithm selected (5 words for SHA1, 7 words for SHA224, 8 words for SHA256, 12 words for SHA384, 16 words for SHA512).

**Bits 25:24 – CHECK[1:0]** Hash Check
Values not listed in table must be considered as “reserved”.

| Value | Name | Description |
|---|---|---|
| 0 | NO_CHECK | No check is performed |
| 1 | CHECK_EHV | Check is performed with expected hash stored in internal expected hash value registers. |
| 2 | CHECK_MESSAGE | Check is performed with expected hash provided after the message. |

**Bit 16 – DUALBUFF** Dual Input Buffer

| Value | Name | Description |
|---|---|---|
| 0 | INACTIVE | SHA_IDATARx and SHA_IODATARx cannot be written during processing of previous block. |
| 1 | ACTIVE | SHA_IDATARx and SHA_IODATARx can be written during processing of previous block when SMOD value = 2. It speeds up the overall runtime of large files. |

**Bits 11:8 – ALGO[3:0]** SHA Algorithm
Values not listed in the table must be considered as “reserved”.

| Value | Name | Description |
|---|---|---|
| 0 | SHA1 | SHA1 algorithm processed |
| 1 | SHA256 | SHA256 algorithm processed |
| 2 | SHA384 | SHA384 algorithm processed |
| 3 | SHA512 | SHA512 algorithm processed |
| 4 | SHA224 | SHA224 algorithm processed |
| 8 | HMAC_SHA1 | HMAC algorithm with SHA1 Hash processed |
| 9 | HMAC_SHA256 | HMAC algorithm with SHA256 Hash processed |
| 10 | HMAC_SHA384 | HMAC algorithm with SHA384 Hash processed |
| 11 | HMAC_SHA512 | HMAC algorithm with SHA512 Hash processed |
| 12 | HMAC_SHA224 | HMAC algorithm with SHA224 Hash processed |
| 13 | Reserved | – |
| 14 | Reserved | – |

**Bit 6 – UIEHV** User Initial or Expected Hash Value Registers

| Value | Description |
|---|---|
| 0 | The SHA algorithm is started with the standard initial values as defined in the FIPS180-2 specification. |
| 1 | The SHA algorithm is started with the user initial hash values stored in the internal register 1 (IR1). If HMAC is configured, UIEHV has no effect (i.e. IR1 is always selected). |

**Bit 5 – UIHV** User Initial Hash Values

| Value | Description |
|---|---|
| 0 | The SHA algorithm is started with the standard initial values as defined in the FIPS180-2 specification. |
| 1 | The SHA algorithm is started with the user initial hash values stored in the internal register 0 (IR0). If HMAC is configured, UIHV has no effect (i.e. IR0 is selected). |

**Bit 4 – PROCDLY** Processing Delay
When SHA1 algorithm is processed, runtime period is either 85 or 209 clock cycles.

When SHA256 or SHA224 algorithm is processed, runtime period is either 72 or 194 clock cycles.

When SHA384 or SHA512 algorithm is processed, runtime period is either 88 or 209 clock cycles.

| Value | Name | Description |
|---|---|---|
| 0 | SHORTEST | SHA processing runtime is the shortest one |
| 1 | LONGEST | SHA processing runtime is the longest one (reduces the SHA bandwidth requirement, reduces the system bus overload) |

**Bits 1:0 – SMOD[1:0]** Start Mode
Values not listed in the table must be considered as “reserved”.

If a DMA transfer is used, configure the SMOD value to 2. See DMA Mode for details.

| Value | Name | Description |
|---|---|---|
| 0 | MANUAL_START | Manual mode |

| Value | Name | Description |
|---|---|---|
| 1 | AUTO_START | Auto mode |
| 2 | IDATAR0_START | SHA_IDATAR0 access only mode (mandatory when DMA is used) |

### 60.5.3 SHA Interrupt Enable Register

**Name:** SHA_IER
**Offset:** 0x10
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CHECKF |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 16 – CHECKF** Check Done Interrupt Enable

**Bit 8 – URAD** Unspecified Register Access Detection Interrupt Enable

**Bit 0 – DATRDY** Data Ready Interrupt Enable

### 60.5.4 SHA Interrupt Disable Register

**Name:** SHA_IDR
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CHECKF |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 16 – CHECKF** Check Done Interrupt Disable

**Bit 8 – URAD** Unspecified Register Access Detection Interrupt Disable

**Bit 0 – DATRDY** Data Ready Interrupt Disable

### 60.5.5 SHA Interrupt Mask Register

**Name:** SHA_IMR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CHECKF |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 16 – CHECKF** Check Done Interrupt Mask

**Bit 8 – URAD** Unspecified Register Access Detection Interrupt Mask

**Bit 0 – DATRDY** Data Ready Interrupt Mask

### 60.5.6 SHA Interrupt Status Register

**Name:** SHA_ISR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CHKST[3:0] | | | | | | | CHECKF |
| Access | R | R | R | R | | | | R |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | URAT[2:0] | | | | | | URAD |
| Access | | R | R | R | | | | R |
| Reset | | 0 | 0 | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | WRDY | | | | DATRDY |
| Access | | | | R | | | | R |
| Reset | | | | 0 | | | | 0 |

**Bits 23:20 – CHKST[3:0]** Check Status (cleared by writing START or SWRST bits in SHA_CR or by reading SHA_IODATARx)
Value 5 indicates identical hash values (expected hash = hash result). Any other value indicates different hash values.

**Bit 16 – CHECKF** Check Done Status (cleared by writing START or SWRST bits in SHA_CR or by reading SHA_IODATARx)

| Value | Description |
|---|---|
| 0 | Hash check has not been computed. |
| 1 | Hash check has been computed, status is available in the CHKST bits. |

**Bits 14:12 – URAT[2:0]** Unspecified Register Access Type (cleared by writing a 1 to SWRST bit in SHA_CR)
Only the last Unspecified Register Access Type is available through the URAT field.

| Value | Name |
|---|---|
| 0 | SHA_IDATAR0 to SHA_IDATAR15 written during the data processing in DMA mode (URAD = 1 and URAT = 0 can occur only if DUALBUFF is cleared in SHA_MR). |
| 1 | Output Data Register read during the data processing. |
| 2 | SHA_MR written during the data processing. |
| 3 | Write-only register read access. |

**Bit 8 – URAD** Unspecified Register Access Detection Status (cleared by writing a 1 to SWRST bit in SHA_CR)

| Value | Description |
|---|---|
| 0 | No unspecified register access has been detected since the last SWRST. |
| 1 | At least one unspecified register access has been detected since the last SWRST. |

**Bit 4 – WRDY** Input Data Register Write Ready

| Value | Description |
|---|---|
| 0 | SHA_IDATAR0 cannot be written |
| 1 | SHA_IDATAR0 can be written |

**Bit 0 – DATRDY** Data Ready (cleared by writing a 1 to bit SWRST or START in SHA_CR, or by reading SHA_IODATARx)

| Value | Description |
|---|---|
| 0 | Output data is not valid. |
| 1 | 512/1024-bit block process is completed. DATRDY is cleared when one of the following conditions is met:<br>• Bit START in SHA_CR is set.<br>• Bit SWRST in SHA_CR is set.<br>• The hash result is read. |

### 60.5.7 SHA Message Size Register

**Name:** SHA_MSR
**Offset:** 0x20
**Reset:** 0x0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | MSGSIZE[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MSGSIZE[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MSGSIZE[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MSGSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – MSGSIZE[31:0]** Message Size
The size in bytes of the message. When MSGSIZE differs from 0, the SHA appends the corresponding value converted in bits after the padding section, as described in the FIPS180-2 specification.

To disable automatic padding, MSGSIZE field must be written to 0.

### 60.5.8 SHA Bytes Count Register

**Name:** SHA_BCR
**Offset:** 0x30
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BYTCNT[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BYTCNT[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BYTCNT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BYTCNT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – BYTCNT[31:0]** Remaining Byte Count Before Auto Padding
When the hash processing starts from the beginning of a message (without preprocessed hash part), BYTCNT must be written with the same value as the MSGSIZE. If a part of the message has been already hashed and the hash does not start from the beginning, BYTCNT must be configured with the number of bytes remaining to process before padding section.

When read, provides the size in bytes of message remaining to be written before the automatic padding starts.

BYTCNT field is automatically updated each time a write occurs in the SHA_IDATARx and SHA_IODATARx.

When BYTCNT reaches 0, the MSGSIZE is converted into bit count and appended at the end of the message after the padding as described in the FIPS180-2 specification.

To disable automatic padding, MSGSIZE and BYTCNT fields must be written to 0.

### 60.5.9 SHA Input Data x Register

**Name:** SHA_IDATARx
**Offset:** 0x40 + x*0x04 [x=0..15]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – IDATA[31:0]** Input Data
The 32-bit Input Data registers allow to load the data block used for hash processing.

These registers are write-only to prevent the input data from being read by another application.

SHA_IDATAR0 corresponds to the first word of the block, SHA_IDATAR15 to the last word of the last block in case SHA algorithm is set to SHA1, SHA224, SHA256 or SHA_IODATA15R to the last word of the block if SHA algorithm is SHA384 or SHA512 (see SHA Input/Output Data Register x).

### 60.5.10 SHA Input/Output Data Register x

**Name:** SHA_IODATARx
**Offset:** 0x80 + x*0x04 [x=0..15]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IODATA[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IODATA[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IODATA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IODATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – IODATA[31:0]** Input/Output Data
These registers can be used to read the resulting message digest and to write the second part of the message block when the SHA algorithm is SHA-384 or SHA-512.

SHA_IODATA0R to SHA_IODATA15R can be written or read but reading these offsets does not return the content of corresponding parts (words) of the message block. Only results from SHA calculation can be read through these registers.

When SHA processing is in progress, these registers return 0x0000.

SHA_IODATAR0 corresponds to the first word of the message digest; SHA_IODATAR4 to the last one in SHA1 mode, SHA_ODATAR6 in SHA224, SHA_IODATAR7 in SHA256, SHA_IODATAR11 in SHA384 or SHA_IODATAR15 in SHA512.

When SHA224 is selected, the content of SHA_ODATAR7 must be ignored.

When SHA384 is selected, the content of SHA_IODATAR12 to SHA_IODATAR15 must be ignored.

# 61. Triple Data Encryption Standard (TDES)

## 61.1 Description

The Triple Data Encryption Standard (TDES) is compliant with the American FIPS (Federal Information Processing Standard) Publication 46-3 specification.

The TDES supports the four different confidentiality modes of operation (ECB, CBC, OFB and CFB), specified in the FIPS (Federal Information Processing Standard) Publication 81 and is compatible with the Peripheral Data Controller channels for all of these modes, minimizing processor intervention for large buffer transfers.

The TDES key is loaded by the software.

The software can write up to three 64-bit keys, each stored in two 32-bit write-only registers, i.e., Key x Word Registers TDES_KEYxWR0 and TDES_KEYxWR1.

The input data (and initialization vector for some modes) are stored in two corresponding 32-bit write-only registers:

- Input Data Registers TDES_IDATAR0 and TDES_IDATAR1
- Initialization Vector Registers TDES_IVR0 and TDES_IVR1

As soon as the initialization vector, the input data and the keys are configured, the encryption/decryption process may be started. Then the encrypted/decrypted data is ready to be read out on the two 32-bit Output Data registers (TDES_ODATARx) or through the DMA channels.

## 61.2 Embedded Characteristics

- Supports Single Data Encryption Standard (DES) and Triple Data Encryption Standard (TDES)
- Compliant with FIPS Publication 46-3, Data Encryption Standard (DES)
- 64-bit Cryptographic Key for TDES
- Two-key or Three-key Algorithms for TDES
- 18-clock Cycles Encryption/Decryption Processing Time for DES
- 50-clock Cycles Encryption/Decryption Processing Time for TDES
- Supports eXtended Tiny Encryption Algorithm (XTEA)
- 128-bit key for XTEA and Programmable Round Number up to 64
- Supports the Four Standard Modes of Operation specified in the FIPS Publication 81, DES Modes of Operation
  - Electronic Code Book (ECB)
  - Cipher Block Chaining (CBC)
  - Cipher Feedback (CFB)
  - Output Feedback (OFB)
- 8-, 16-, 32- and 64-bit Data Sizes Possible in CFB Mode
- Last Output Data Mode Allowing Optimized Message (Data) Authentication Code (MAC) Generation
- Connection to DMA Optimizes Data Transfers for all Operating Modes

## 61.3 Product Dependencies

### 61.3.1 Power Management

The TDES may be clocked through the Power Management Controller (PMC), so the programmer must first configure the PMC to enable the TDES clock.

### 61.3.2 Interrupt Sources

The TDES interface has an interrupt line connected to the Interrupt Controller. In order to handle interrupts, the Interrupt Controller must be programmed before configuring the TDES.

## 61.4 Functional Description

The Data Encryption Standard (DES) and the Triple Data Encryption Algorithm (TDES) specify FIPS-approved cryptographic algorithms that can be used to protect electronic data. The TDES bit in the TDES Mode Register (TDES_MR) is used to select either the single DES or the Triple DES mode.

Encryption (enciphering) converts data to an unintelligible form called ciphertext. Decrypting (deciphering) the ciphertext converts the data back into its original form, called plaintext. The CIPHER bit in TDES_MR is used to choose between encryption and decryption.

A DES is capable of using cryptographic keys of 64 bits to encrypt and decrypt data in blocks of 64 bits. This 64-bit key is defined in the Key 1 Registers (TDES_KEY1WRx ).

A TDES key consists of three DES keys, which is also referred to as a key bundle. These three 64-bit keys are defined, respectively, in the Key 1, 2 and 3 Registers (TDES_KEY1WRx, TDES_KEY2WRx and TDES_KEY3WRx). In Triple DES mode (TDESMOD = 1 in TDES_MR), the KEYMOD bit in TDES_MR is used to choose between a two- and a three-key algorithm, as summarized in the table below.

**Table 61-1. TDES Algorithms Summary**

| Algorithm | Mode | Data Processing Sequence Steps | | |
|---|---|---|---|---|
| | | First | Second | Third |
| Three-key | Encryption | Encryption with Key 1 | Decryption with Key 2 | Encryption with Key 3 |
| | Decryption | Decryption with Key 3 | Encryption with Key 2 | Decryption with Key 1 |
| Two-key | Encryption | Encryption with Key 1 | Decryption with Key 2 | Encryption with Key 1 |
| | Decryption | Decryption with Key 1 | Encryption with Key 2 | Decryption with Key 1 |

The input to the encryption processes of the CBC, CFB, and OFB modes includes, in addition to the plaintext, a 64-bit data block called the initialization vector (IV), which must be set in the Initialization Vector Registers (TDES_IVRx). The initialization vector is used in an initial step in the encryption of a message and in the corresponding decryption of the message.

The XTEA algorithm can be used instead of DES/TDES by configuring the TDESMOD field in TDES_MR with the appropriate value 0x2. An XTEA key consists of a 128-bit key. They are defined in the Key 1 and 2 Registers.

The number of rounds of XTEA is defined in TDES_XTEA_RNDR and can be programmed up to 64 (1 round = 2 Feistel network rounds).

All the start and operating modes of the TDES algorithm can be applied to the XTEA algorithm.

### 61.4.1 Operating Modes

The TDES supports the following operating modes:

- ECB—Electronic Code Book
- CBC—Cipher Block Chaining
- OFB—Output Feedback
- CFB—Cipher Feedback
  - CFB8 (CFB where the length of the data segment is 8 bits)
  - CFB16 (CFB where the length of the data segment is 16 bits)
  - CFB32 (CFB where the length of the data segment is 32 bits)
  - CFB64 (CFB where the length of the data segment is 64 bits)

The data pre-processing, post-processing and data chaining for each mode are automatically performed. Refer to the FIPS Publication 81 for more complete information.

These modes are selected by setting the OPMOD field in TDES_MR.

In CFB mode, four data sizes are possible (8, 16, 32 and 64 bits), configurable by means of the CFBS field in TDES_MR (see TDES Mode Register).

### 61.4.2 Start Modes

The SMOD field in TDES_MR selects the Encryption (or Decryption) start mode.

#### 61.4.2.1 Manual Mode

The sequence is as follows:

1. Write the TDES_MR register with all required fields, including but not limited to SMOD and OPMOD.
2. Write the 64-bit key(s) in the different Key Registers (TDES_KEYxWRx), depending on whether one, two or three keys are required.
3. Write the initialization vector (or counter) in the Initialization Vector Registers (TDES_IVRx).
   **Note:** The Initialization Vector Registers concern all modes except ECB.
4. Set the bit DATRDY (Data Ready) in the TDES Interrupt Enable register (TDES_IER), depending on whether an interrupt is required or not at the end of processing.
5. Write the data to be encrypted/decrypted in the authorized Input Data Registers (see the table below).
   **Note:** In 32-, 16- and 8-bit CFB modes, writing to TDES_IDATAR1 is not allowed and may lead to processing errors.
6. Set the START bit in the TDES Control Register (TDES_CR) to begin the encryption or decryption process.
7. When the processing completes, the bit DATRDY in the TDES Interrupt Status Register (TDES_ISR) rises. If an interrupt has been enabled by setting the bit DATRDY in TDES_IER, the interrupt line of the TDES is activated.
8. When the software reads one of the Output Data Registers (TDES_ODATARx), the DATRDY bit is automatically cleared.

**Table 61-2. Authorized Input Data Registers**

| Operating Mode | Input Data Registers to Write |
|---|---|
| ECB | All |

| .....continued | |
|---|---|
| **Operating Mode** | **Input Data Registers to Write** |
| CBC | All |
| OFB | All |
| CFB 64-bit | All |
| CFB 32-bit | TDES_IDATAR0 |
| CFB 16-bit | TDES_IDATAR0 |
| CFB 8-bit | TDES_IDATAR0 |

#### 61.4.2.2 Auto Mode

The Auto Mode is similar to the Manual Mode, except that as soon as the correct number of Input Data registers is written, processing is automatically started without any action in TDES_CR.

#### 61.4.2.3 DMA Mode

The DMA Controller can be used in association with the TDES to perform an encryption/decryption of a buffer without any action by the software during processing.

The SMOD field of TDES_MR must be set to 0x2 and the DMA must be configured with non-incremental addresses.

For all operating modes except CBC-MAC (TDES_MR.LOD=1), 2 DMA channels must be programmed (transmit and receive). In CBC-MAC, only 1 transmit channel must be programmed.

The start address of any transfer descriptor must be set in TDES_IDATAR0.

The DMA chunk size configuration depends on the TDES mode of operation and is listed in the table below.

When writing data to TDES with the first DMA channel, data will be fetched from a memory buffer (source data). It is recommended to configure the size of source data to “words” even for CFB modes. On the contrary, the destination data size depends on the mode of operation. When reading data from the TDES with the second DMA channel, the source data is the data read from TDES and data destination is the memory buffer. In this case, source data size depends on the TDES mode of operation and is listed in the table below.

**Table 61-3. DMA Data Transfer Type for the Different Operating Modes**

| Operating Mode | Chunk Size | Destination/Source Data Transfer Type |
|---|---|---|
| ECB | 1 | Word |
| CBC | 1 | Word |
| OFB | 1 | Word |
| CFB 64-bit | 1 | Word |
| CFB 32-bit | 1 | Word |
| CFB 16-bit | 1 | Half-word |
| CFB 8-bit | 1 | Byte |

### 61.4.3 Last Output Data Mode (CBC-MAC)

This mode is used to generate cryptographic checksums on data (MAC) using a CBC-MAC or a CFB encryption algorithm (refer to *FIPS Publication 81 Appendix F*).

The CMAC algorithm is a variant of CBC-MAC with post-processing requiring one-block encryption in ECB mode. Thus CBC-MAC is useful to accelerate CMAC.

After each end of encryption/decryption, the output data is available either on the output data registers for Manual and Auto modes or at the address specified in the receive buffer pointer for DMA mode (See Last Output Data Mode Behavior versus Start Modes).

The Last Output Data bit (LOD) in TDES_MR can be used to retrieve only the last data of several encryption/decryption processes.

This data is only available on the Output Data Registers (TDES_ODATARx).

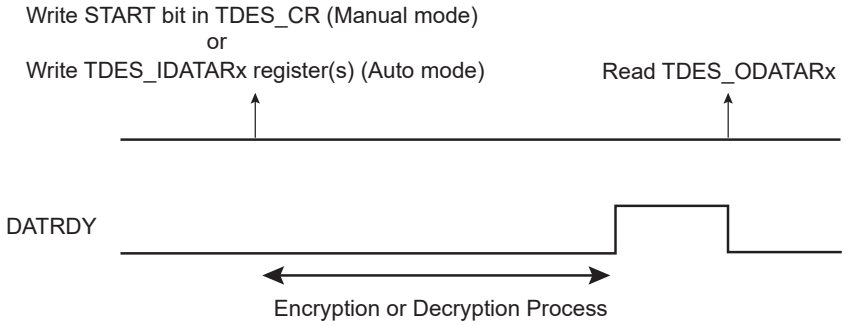
Therefore, there is no need to define a read buffer in DMA mode.

#### 61.4.3.1 Manual and Auto Modes

##### 61.4.3.1.1 TDES_MR.LOD = 0

The DATRDY flag is cleared when at least one of the Output Data Registers is read. See the figure below.

**Figure 61-1. Manual and Auto Modes with LOD = 0**

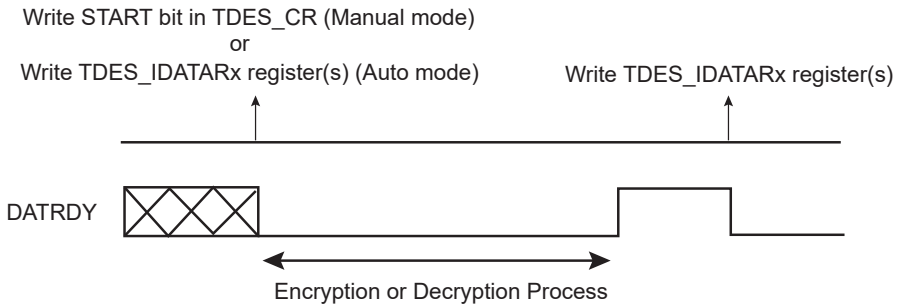



If the user does not want to read the output data registers between each encryption/decryption, the DATRDY flag will not be cleared. If the DATRDY flag is not cleared, the user will not be informed of the end of the encryptions/decryptions that follow.

##### 61.4.3.1.2 TDES_MR.LOD = 1

The DATRDY flag is cleared when at least one Input Data Register is written, before the start of a new transfer. See the figure below. No further Output Data Register reads are necessary between consecutive encryptions/decryptions.

**Figure 61-2. Manual and Auto Modes with LOD = 1**



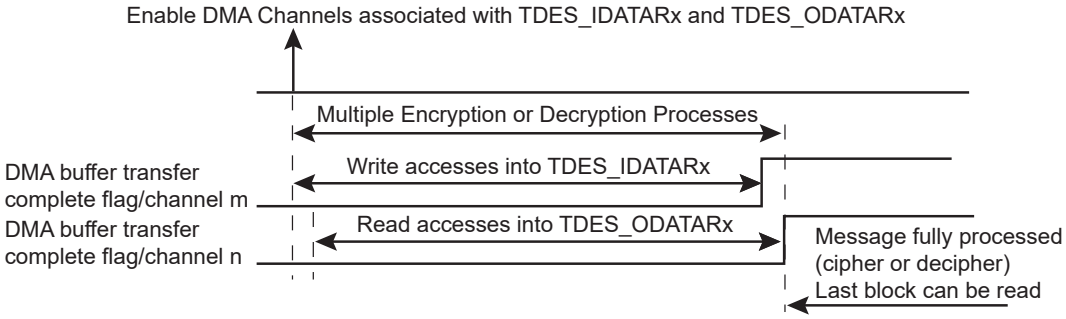
### 61.4.3.2 DMA Mode

#### 61.4.3.2.1 TDES_MR.LOD = 0

This mode may be used for all TDES operating modes except CBC-MAC where LOD = 1 mode is recommended.

The end of the encryption/decryption is indicated by the end of DMA transfer associated to TDES_ODATARx (see the figure below). Two DMA channels are required: one for writing message blocks to TDES_IDATARx and one to obtain the result from TDES_ODATARx.

**Figure 61-3. DMA Transfer with LOD = 0**




#### 61.4.3.2.2 TDES_MR.LOD = 1

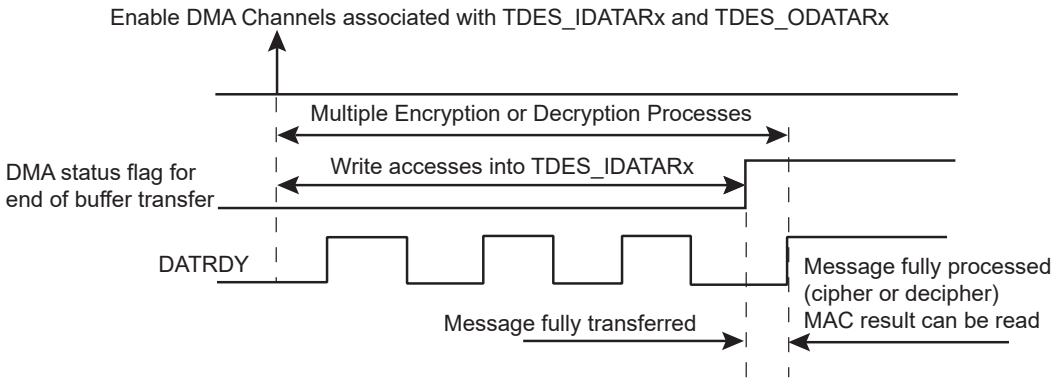
This mode is optimized to process the TDES CBC-MAC operating mode.

The user must first wait for the DMA buffer transfer complete flag, then for the flag DATRDY to rise to ensure that the encryption/decryption is completed (see the figure below).

The DMA receive channel must not be used. Prior to reading the CBC-MAC result, TDES_MR.SMOD must be written to ‘0’. To restart a CBC-MAC on a new buffer, TDES_MR.SMOD must be written to ‘2’.

The output data is only available on TDES_ODATARx.

**Figure 61-4. DMA Transfer with LOD = 1**




The table below summarizes the different cases.

**Table 61-4. Last Output Data Mode Behavior versus Start Modes**

| Sequence | Manual and Auto Modes | | DMA Transfer | |
|---|---|---|---|---|
| | LOD = 0 | LOD = 1 | LOD = 0 | LOD = 1 |
| DATRDY Flag Clearing Condition (1) | At least one Output Data Register must be read | At least one Input Data Register must be written | Not used | Managed by the DMA |

.....continued

| Sequence | Manual and Auto Modes | | DMA Transfer | |
|---|---|---|---|---|
| | LOD = 0 | LOD = 1 | LOD = 0 | LOD = 1 |
| End of Encryption/ Decryption | DATRDY | DATRDY | 2 DMA Buffer transfer complete flags (channel m and channel n) | DMA buffer transfer complete flag, then TDES DATRDY flag |
| Encrypted/ Decrypted Data Result Location | In the Output Data Registers | In the Output Data Registers | Not available | In the Output Data Registers |

**Note:** Depending on the mode, there are other ways of clearing the DATRDY flag. See TDES Interrupt Status Register.

**WARNING** In DMA mode, reading to the Output Data registers before the last data transfer may lead to unpredictable results.

### 61.4.4 Security Features

#### 61.4.4.1 Unspecified Register Access Detection

When an unspecified register access occurs, the URAD bit in TDES_ISR is set. Its source is then reported in the Unspecified Register Access Type field (URAT). Only the last unspecified register access is available through the URAT field.

Several kinds of unspecified register accesses can occur:

- Input Data Register written during the data processing in DMA mode
- Output Data Register read during the data processing
- Mode Register written during the data processing
- Write-only register read access

The URAD bit and the URAT field can only be reset by the SWRST bit in TDES_CR.

## 61.5 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | TDES_CR | 7:0 | | | | | | | | START |
| | | 15:8 | | | | | | | | SWRST |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | TDES_MR | 7:0 | | | | KEYMOD | | TDESMOD[1:0] | | CIPHER |
| | | 15:8 | LOD | | OPMOD[1:0] | | | | SMOD[1:0] | |
| | | 23:16 | | | | | | | CFBS[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x08 ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | TDES_IER | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | URAD |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | TDES_IDR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | URAD |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | TDES_IMR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | URAD |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | TDES_ISR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | URAT[1:0] | | | | | URAD |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | TDES_KEY1WR0 | 7:0 | KEY1W[7:0] | | | | | | | |
| | | 15:8 | KEY1W[15:8] | | | | | | | |
| | | 23:16 | KEY1W[23:16] | | | | | | | |
| | | 31:24 | KEY1W[31:24] | | | | | | | |
| 0x24 | TDES_KEY1WR1 | 7:0 | KEY1W[7:0] | | | | | | | |
| | | 15:8 | KEY1W[15:8] | | | | | | | |
| | | 23:16 | KEY1W[23:16] | | | | | | | |
| | | 31:24 | KEY1W[31:24] | | | | | | | |
| 0x28 | TDES_KEY2WR0 | 7:0 | KEY2W[7:0] | | | | | | | |
| | | 15:8 | KEY2W[15:8] | | | | | | | |
| | | 23:16 | KEY2W[23:16] | | | | | | | |
| | | 31:24 | KEY2W[31:24] | | | | | | | |
| 0x2C | TDES_KEY2WR1 | 7:0 | KEY2W[7:0] | | | | | | | |
| | | 15:8 | KEY2W[15:8] | | | | | | | |
| | | 23:16 | KEY2W[23:16] | | | | | | | |
| | | 31:24 | KEY2W[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x30 | TDES_KEY3WR0 | 7:0 | KEY3W[7:0] | | | | | | | |
| | | 15:8 | KEY3W[15:8] | | | | | | | |
| | | 23:16 | KEY3W[23:16] | | | | | | | |
| | | 31:24 | KEY3W[31:24] | | | | | | | |
| 0x34 | TDES_KEY3WR1 | 7:0 | KEY3W[7:0] | | | | | | | |
| | | 15:8 | KEY3W[15:8] | | | | | | | |
| | | 23:16 | KEY3W[23:16] | | | | | | | |
| | | 31:24 | KEY3W[31:24] | | | | | | | |
| 0x38 ... 0x3F | Reserved | | | | | | | | | |
| 0x40 | TDES_IDATAR0 | 7:0 | IDATA[7:0] | | | | | | | |
| | | 15:8 | IDATA[15:8] | | | | | | | |
| | | 23:16 | IDATA[23:16] | | | | | | | |
| | | 31:24 | IDATA[31:24] | | | | | | | |
| 0x44 | TDES_IDATAR1 | 7:0 | IDATA[7:0] | | | | | | | |
| | | 15:8 | IDATA[15:8] | | | | | | | |
| | | 23:16 | IDATA[23:16] | | | | | | | |
| | | 31:24 | IDATA[31:24] | | | | | | | |
| 0x48 ... 0x4F | Reserved | | | | | | | | | |
| 0x50 | TDES_ODATAR0 | 7:0 | ODATA[7:0] | | | | | | | |
| | | 15:8 | ODATA[15:8] | | | | | | | |
| | | 23:16 | ODATA[23:16] | | | | | | | |
| | | 31:24 | ODATA[31:24] | | | | | | | |
| 0x54 | TDES_ODATAR1 | 7:0 | ODATA[7:0] | | | | | | | |
| | | 15:8 | ODATA[15:8] | | | | | | | |
| | | 23:16 | ODATA[23:16] | | | | | | | |
| | | 31:24 | ODATA[31:24] | | | | | | | |
| 0x58 ... 0x5F | Reserved | | | | | | | | | |
| 0x60 | TDES_IVR0 | 7:0 | IV[7:0] | | | | | | | |
| | | 15:8 | IV[15:8] | | | | | | | |
| | | 23:16 | IV[23:16] | | | | | | | |
| | | 31:24 | IV[31:24] | | | | | | | |
| 0x64 | TDES_IVR1 | 7:0 | IV[7:0] | | | | | | | |
| | | 15:8 | IV[15:8] | | | | | | | |
| | | 23:16 | IV[23:16] | | | | | | | |
| | | 31:24 | IV[31:24] | | | | | | | |
| 0x68 ... 0x6F | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x70 | TDES_XTEA_RNDR | 7:0 | | | XTEA_RNDS[5:0] | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 61.5.1 TDES Control Register

**Name:** TDES_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SWRST |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | START |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 8 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Resets the TDES. A software triggered hardware reset of the TDES interface is performed. |

**Bit 0 – START** Start Processing

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Starts Manual encryption/decryption process. |

### 61.5.2 TDES Mode Register

**Name:** TDES_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CFBS[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LOD | | OPMOD[1:0] | | | | SMOD[1:0] | |
| Access | R/W | | R/W | R/W | | | R/W | R/W |
| Reset | 0 | | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | KEYMOD | | TDESMOD[1:0] | | CIPHER |
| Access | | | | R/W | | R/W | R/W | R/W |
| Reset | | | | 0 | | 0 | 0 | 0 |

**Bits 17:16 – CFBS[1:0]** Cipher Feedback Data Size

| Value | Name | Description |
|---|---|---|
| 0 | SIZE_64BIT | 64-bit |
| 1 | SIZE_32BIT | 32-bit |
| 2 | SIZE_16BIT | 16-bit |
| 3 | SIZE_8BIT | 8-bit |

**Bit 15 – LOD** Last Output Data Mode

**WARNING** In DMA mode, reading to the Output Data registers before the last data encryption/decryption process may lead to unpredictable result.

| Value | Description |
|---|---|
| 0 | No effect.<br>After each end of encryption/decryption, the output data is available either on the output data registers (Manual and Auto modes) .<br>In Manual and Auto modes, the DATRDY flag is cleared when at least one of the Output Data registers is read. |

| Value | Description |
|---|---|
| 1 | The DATRDY flag is cleared when at least one of the Input Data Registers is written.<br>No more Output Data Register reads are necessary between consecutive encryptions/ decryptions (see Last Output Data Mode). |

**Bits 13:12 – OPMOD[1:0]** Operating Mode
For CBC-MAC operating mode, set OPMOD to CBC and LOD to 1.

| Value | Name | Description |
|---|---|---|
| 0 | ECB | Electronic Code Book mode |
| 1 | CBC | Cipher Block Chaining mode |
| 2 | OFB | Output Feedback mode |
| 3 | CFB | Cipher Feedback mode |

**Bits 9:8 – SMOD[1:0]** Start Mode
If a DMA transfer is used, 0x2 must be configured. See DMA Mode for more details.

| Value | Name | Description |
|---|---|---|
| 0 | MANUAL_START | Manual Mode |
| 1 | AUTO_START | Auto Mode |
| 2 | IDATAR0_START | TDES_IDATAR0 accesses only Auto Mode |

**Bit 4 – KEYMOD** Key Mode

| Value | Description |
|---|---|
| 0 | Three-key algorithm is selected. |
| 1 | Two-key algorithm is selected. There is no need to write Key 3 Registers TDES_KEY3WRx. |

**Bits 2:1 – TDESMOD[1:0]** ALGORITHM Mode
Values which are not listed in the table must be considered as “reserved”.

| Value | Name | Description |
|---|---|---|
| 0 | SINGLE_DES | Single DES processing using Key 1 Registers |
| 1 | TRIPLE_DES | Triple DES processing using Key 1, Key 2 and Key 3 Registers |
| 2 | XTEA | XTEA processing using Key 1 and Key 2 Registers |

**Bit 0 – CIPHER** Processing Mode

| Value | Name | Description |
|---|---|---|
| 0 | DECRYPT | Decrypts data. |
| 1 | ENCRYPT | Encrypts data. |

### 61.5.3 TDES Interrupt Enable Register

**Name:** TDES_IER
**Offset:** 0x10
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 8 – URAD** Unspecified Register Access Detection Interrupt Enable

**Bit 0 – DATRDY** Data Ready Interrupt Enable

### 61.5.4 TDES Interrupt Disable Register

**Name:** TDES_IDR
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 8 – URAD** Unspecified Register Access Detection Interrupt Disable

**Bit 0 – DATRDY** Data Ready Interrupt Disable

### 61.5.5 TDES Interrupt Mask Register

**Name:** TDES_IMR
**Offset:** 0x18
**Reset:** 0x000000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | W |
| Reset | | | | | | | | 0 |

**Bit 8 – URAD** Unspecified Register Access Detection Interrupt Mask

**Bit 0 – DATRDY** Data Ready Interrupt Mask

### 61.5.6 TDES Interrupt Status Register

**Name:** TDES_ISR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | URAT[1:0] | | | | | URAD |
| Access | | | R | R | | | | R |
| Reset | | | 0 | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 13:12 – URAT[1:0]** Unspecified Register Access (cleared by setting bit TDES_CR.SWRST)
Only the last Unspecified Register Access Type is available through the URAT field.

| Value | Name | Description |
|---|---|---|
| 0 | IDR_WR_PROCESSING | Input Data Register written during data processing when SMOD = 0x2 mode. |
| 1 | ODR_RD_PROCESSING | Output Data Register read during data processing. |
| 2 | MR_WR_PROCESSING | Mode Register written during data processing. |
| 3 | WOR_RD_ACCESS | Write-only register read access. |

**Bit 8 – URAD** Unspecified Register Access Detection Status (cleared by setting bit TDES_CR.SWRST)

| Value | Description |
|---|---|
| 0 | No unspecified register access has been detected since the last write of bit TDES_CR.SWRST. |
| 1 | At least one unspecified register access has been detected since the last write of bit TDES_CR.SWRST. |

**Bit 0 – DATRDY** Data Ready (cleared by setting bit START or bit SWRST in TDES_CR or by reading TDES_ODATARx)
If TDES_MR.LOD = 1: In Manual and Auto modes, the DATRDY flag can also be cleared by writing at least one TDES_IDATARx.

| Value | Description |
|---|---|
| 0 | Output data is not valid. |
| 1 | Encryption or decryption process is completed. |

### 61.5.7 TDES Key 1 Word Register x

**Name:** TDES_KEY1WRx
**Offset:** 0x20 + x*0x04 [x=0..1]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY1W[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEY1W[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | KEY1W[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | KEY1W[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 31:0 – KEY1W[31:0]** Key 1 Word
The two 32-bit Key 1 Word registers are used to set the 64-bit cryptographic key used for encryption/decryption.

KEY1W0 refers to the first word of the key and KEY1W1 to the last one.

These registers are write-only to prevent the key from being read by another application.

In XTEA mode, the key is defined on 128 bits. These registers contain the 64 LSB bits of the encryption/decryption key.

### 61.5.8 TDES Key 2 Word Register x

**Name:** TDES_KEY2WRx
**Offset:** 0x28 + x*0x04 [x=0..1]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY2W[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEY2W[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | KEY2W[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | KEY2W[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 31:0 – KEY2W[31:0]** Key 2 Word
The two 32-bit Key 2 Word registers are used to set the 64-bit cryptographic key used for encryption/decryption.

KEY2W0 refers to the first word of the key and KEY2W1 to the last one.

These registers are write-only to prevent the key from being read by another application.

TDES_KEY2WRx registers are not used in DES mode.

In XTEA mode, the key is defined on 128 bits. These registers contain the 64 MSB bits of the encryption/decryption key.

### 61.5.9 TDES Key 3 Word Register x

**Name:** TDES_KEY3WRx
**Offset:** 0x30 + x*0x04 [x=0..1]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY3W[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEY3W[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | KEY3W[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | KEY3W[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 31:0 – KEY3W[31:0]** Key 3 Word
The two 32-bit Key 3 Word registers are used to set the 64-bit cryptographic key used for encryption/decryption.

KEY3W0 refers to the first word of the key and KEY3W1 to the last one.

These registers are write-only to prevent the key from being read by another application.

TDES_KEY3WRx registers are not used in DES mode, TDES with two-key algorithm selected and XTEA mode.

### 61.5.10 TDES Input Data Register x

**Name:** TDES_IDATARx
**Offset:** 0x40 + x*0x04 [x=0..1]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 31:0 – IDATA[31:0]** Input Data
The two 32-bit Input Data registers are used to set the 64-bit data block used for encryption/decryption.

IDATA0 refers to the first word of the data to be encrypted/decrypted, and IDATA1 to the last one.

These registers are write-only to prevent the input data from being read by another application.

### 61.5.11 TDES Output Data Register x

**Name:** TDES_ODATARx
**Offset:** 0x50 + x*0x04 [x=0..1]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ODATA[31:0]** Output Data
The two 32-bit Output Data registers contain the 64-bit data block which has been encrypted/decrypted.

ODATA1 refers to the first word, ODATA2 to the last one.

### 61.5.12 TDES Initialization Vector Register x

**Name:** TDES_IVRx
**Offset:** 0x60 + x*0x04 [x=0..1]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IV[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IV[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IV[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IV[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 31:0 – IV[31:0]** Initialization Vector
The two 32-bit Initialization Vector registers are used to set the 64-bit initialization vector data block, which is used by some modes of operation as an additional initial input.

IV1 refers to the first word of the Initialization Vector, IV2 to the last one.

These registers are write-only to prevent the Initialization Vector from being read by another application.

These registers are not used for the ECB mode and must not be written.

### 61.5.13 TDES XTEA Rounds Register

**Name:** TDES_XTEA_RNDR
**Offset:** 0x70
**Reset:** 0x000000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | XTEA_RNDS[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:0 – XTEA_RNDS[5:0]** Number of Rounds
This 6-bit field is used to define the number of complete rounds (1 complete round = 2 Feistel rounds) processed in XTEA algorithm.

The value of XTEA_RNDS has no effect if the TDESMOD field in TDES_MR is set to 0x0 or 0x1.

0x00 corresponds to 1 complete round, 0x01 corresponds to 2 complete rounds, etc.

# 62. True Random Number Generator (TRNG)

## 62.1 Description

The True Random Number Generator (TRNG) passes the American *NIST Special Publication 800-22 (A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications) and the Diehard Suite of Tests.*

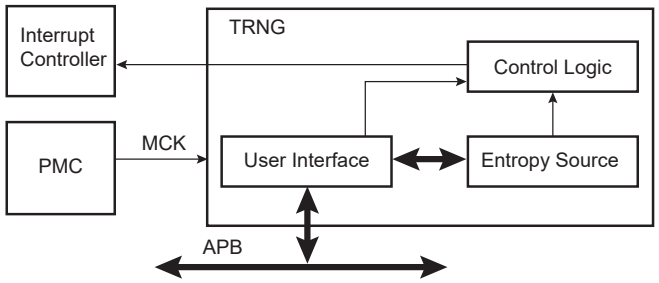
The TRNG may be used as an entropy source for seeding an NIST approved DRNG (Deterministic RNG) as required by FIPS PUB 140-2 and 140-3.

## 62.2 Embedded Characteristics

- Passes *NIST Special Publication 800-22 Test Suite*
- Passes *Diehard Suite of Tests*
- May be Used as Entropy Source for seeding a NIST-approved DRNG (Deterministic RNG) as required by FIPS PUB 140-2 and 140-3
- Provides a 32-bit Random Number Every 84 Clock Cycles

## 62.3 Block Diagram

**Figure 62-1. TRNG Block Diagram**




## 62.4 Product Dependencies

### 62.4.1 Power Management

The TRNG interface may be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the TRNG user interface clock. The user interface clock is independent from any clock that may be used in the entropy source logic circuitry. The source of entropy can be enabled before enabling the user interface clock.

### 62.4.2 Interrupt Sources

The TRNG interface has an interrupt line connected to the Interrupt Controller. In order to handle interrupts, the Interrupt Controller must be programmed before configuring the TRNG.

## 62.5 Functional Description

As soon as the TRNG is enabled in the Control register (TRNG_CR), the generator provides one 32-bit random value every 84 clock cycles.

The TRNG interrupt line can be enabled in the Interrupt Enable register (TRNG_IER), and disabled in the Interrupt Disable register (TRNG_IDR). This interrupt is set when a new random value is available and is cleared when the Status register (TRNG_ISR) is read. The flag TRNG_ISR.DATRDY is set when the random data is ready to be read out on the 32-bit Output Data register (TRNG_ODATA).

## 62.6 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | TRNG_CR | 7:0 | | | | | | | | ENABLE |
| | | 15:8 | WAKEY[7:0] | | | | | | | |
| | | 23:16 | WAKEY[15:8] | | | | | | | |
| | | 31:24 | WAKEY[23:16] | | | | | | | |
| 0x04 ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | TRNG_IER | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | TRNG_IDR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | TRNG_IMR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | TRNG_ISR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 ... 0x4F | Reserved | | | | | | | | | |
| 0x50 | TRNG_ODATA | 7:0 | ODATA[7:0] | | | | | | | |
| | | 15:8 | ODATA[15:8] | | | | | | | |
| | | 23:16 | ODATA[23:16] | | | | | | | |
| | | 31:24 | ODATA[31:24] | | | | | | | |

### 62.6.1 TRNG Control Register

**Name:** TRNG_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WAKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WAKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WAKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ENABLE |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bits 31:8 – WAKEY[23:0]** Register Write Access Key

| Value | Name | Description |
|---|---|---|
| 0x524E47 | PASSWD | Writing any other value in this field aborts the write operation. |

**Bit 0 – ENABLE** Enable TRNG to Provide Random Values

| Value | Description |
|---|---|
| 0 | Disables the TRNG if 0x524E47 (“RNG” in ASCII) is written in WAKEY field at the same time. |
| 1 | Enables the TRNG if 0x524E47 (“RNG” in ASCII) is written in WAKEY field at the same time. |

### 62.6.2 TRNG Interrupt Enable Register

**Name:** TRNG_IER
**Offset:** 0x10
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

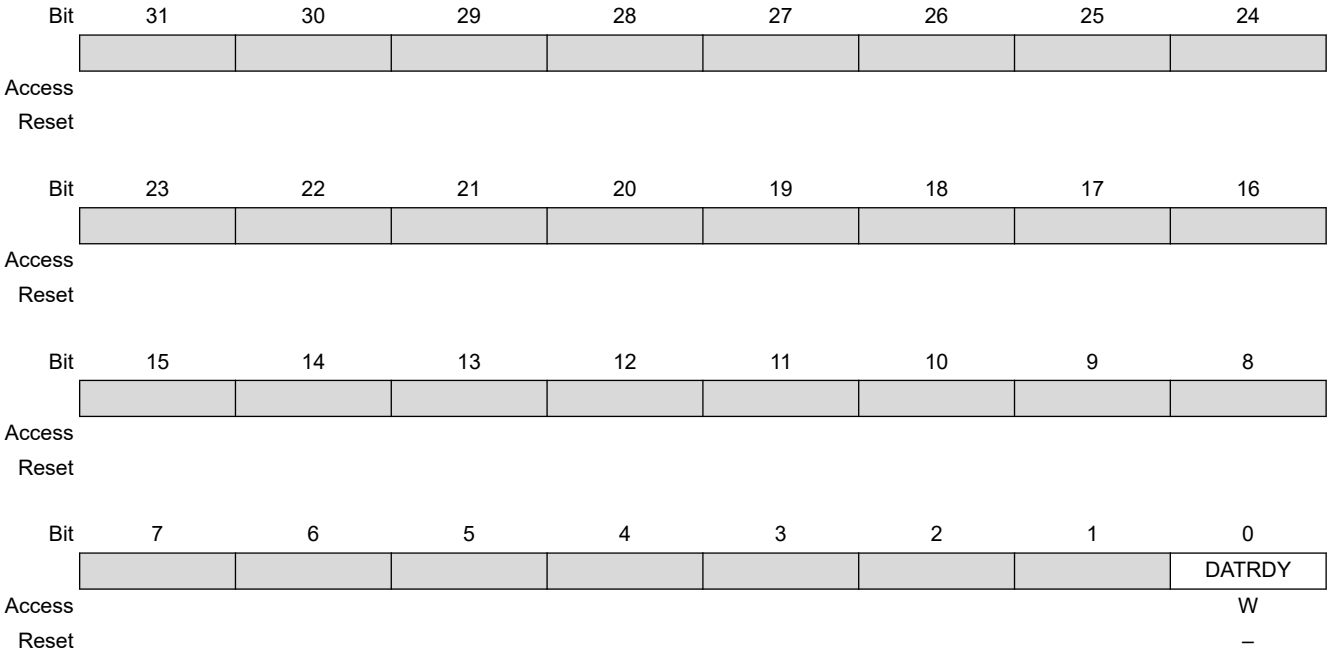
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – DATRDY** Data Ready Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding interrupt. |

### 62.6.3 TRNG Interrupt Disable Register

**Name:** TRNG_IDR
**Offset:** 0x14
**Reset:** –
**Property:** Write-only




**Bit 0 – DATRDY** Data Ready Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding interrupt. |

### 62.6.4 TRNG Interrupt Mask Register

**Name:** TRNG_IMR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 0 – DATRDY** Data Ready Interrupt Mask

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is not enabled. |
| 1 | The corresponding interrupt is enabled. |

### 62.6.5 TRNG Interrupt Status Register

**Name:** TRNG_ISR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 0 – DATRDY** Data Ready (cleared on read)

| Value | Description |
|---|---|
| 0 | Output data is not valid or TRNG is disabled. |
| 1 | New random value is completed since the last read of TRNG_ODATA. |

### 62.6.6 TRNG Output Data Register

**Name:** TRNG_ODATA
**Offset:** 0x50
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ODATA[31:0]** Output Data
The 32-bit Output Data register contains the 32-bit random data.

# 63. Analog Comparator Controller (ACC)

## 63.1 Description

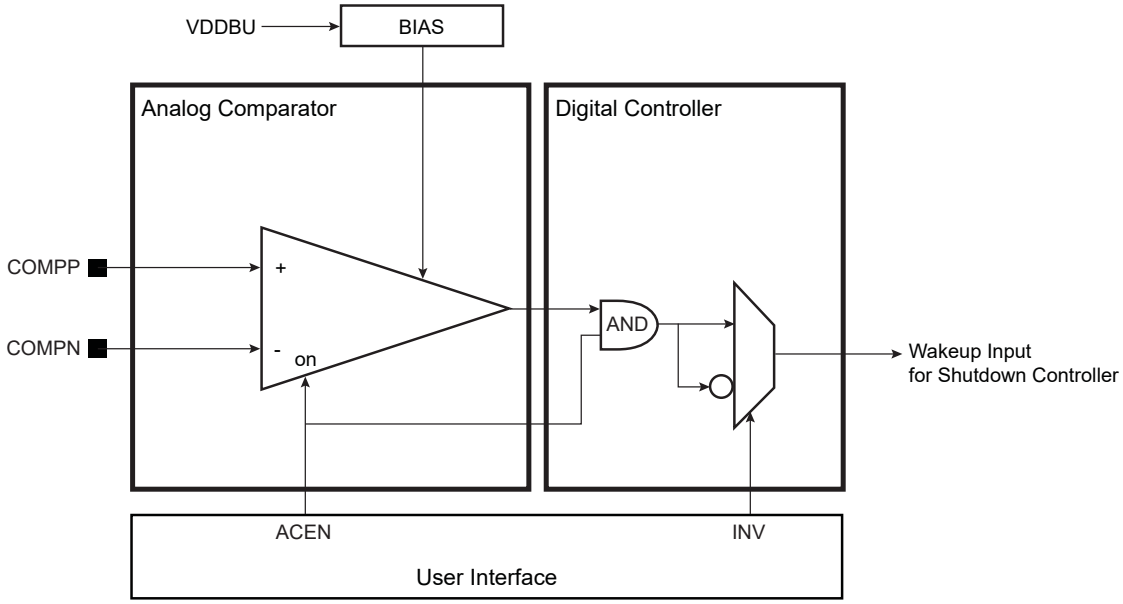
The Analog Comparator Controller (ACC) controls the analog comparator in order to provide an additional source of wakeup when the system wakes up from Wait mode.

## 63.2 Embedded Characteristics

- Source of Wakeup When System Wakes Up from Wait Mode and ULP1 Mode

## 63.3 Block Diagram

**Figure 63-1. Analog Comparator Controller Block Diagram**




## 63.4 Signal Description

**Table 63-1. ACC Signal Description**

| Pin Name | Description | Type |
|---|---|---|
| COMPP, COMPN | External analog data inputs | Input |

## 63.5 Product Dependencies

### 63.5.1 I/O Lines

The analog input pins (COMPP and COMPN) are not multiplexed with digital functions (PIO) on the I/O line.

### 63.5.2 Power Management

By clearing the ACEN bit in the ACC Mode Register (ACC_MR), the analog comparator power consumption is reduced to current leakage only.

## 63.6 Functional Description

### 63.6.1 Description

The analog comparator is enabled by writing a one to the ACEN bit in the ACC Mode Register (ACC_MR) and the polarity of the comparator output can be configured with bit ACC_MR.INV.

The ACC registers are listed in the Register Summary.

### 63.6.2 Register Write Protection

To prevent any single software error from corrupting ACC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the ACC Write Protection Mode Register (ACC_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the ACC Write Protection Status Register (ACC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the ACC_WPSR register.

The following registers can be write-protected:

- ACC Mode Register

## 63.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | ACC_CR | 7:0 | | | | | | | | SWRST |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | ACC_MR | 7:0 | | | | | | | | |
| | | 15:8 | | | | INV | | | | ACEN |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | ACC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | ACC_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 63.7.1 ACC Control Register

**Name:** ACC_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SWRST |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the module. |

### 63.7.2 ACC Mode Register

**Name:** ACC_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the ACC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | INV | | | | ACEN |
| Access | | | | R/W | | | | R/W |
| Reset | | | | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 12 – INV** Invert Comparator Output
0 (DIS): Analog comparator output is directly processed.

1 (EN): Analog comparator output is inverted prior to being processed.

**Bit 8 – ACEN** Analog Comparator Enable
0 (DIS): Analog comparator disabled.

1 (EN): Analog comparator enabled.

### 63.7.3 ACC Write Protection Mode Register

**Name:** ACC_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x414343 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit.<br>Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
Refer to Register Write Protection for the list of registers that can be write-protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x414343 (“ACC” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x414343 (“ACC” in ASCII). |

### 63.7.4 ACC Write Protection Status Register

**Name:** ACC_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of ACC_WPSR. |
| 1 | A write protection violation (WPEN = 1) has occurred since the last read of ACC_WPSR. |

# 64. Security Module (SECUMOD)

## 64.1 Description

The Security Module (SECUMOD) features different levels of security depending on the device reference. This section describes the protections embedded in the SECUMOD available on all SAMA5D2 devices.

This module embeds the secure memories (5 Kbytes of SRAM and a 256-bit register bank) dedicated to the storage of sensitive data. These memories are scrambled with a programmable 32-bit key.

When a fault is detected, regardless of the source, a clear signal can be sent automatically to the secure memories and clear their contents.

For information specific to dynamic tamper protection (PIOBU), refer to the document “SAMA5D2 External Tamper Protections”(document no. 44095).

For information specific to temperature, voltage and frequency monitoring for SAMA5D23 and SAMA5D28, refer to the document “SAMA5D23 and SAMA5D28 Environmental Monitors” (document no. 44036).

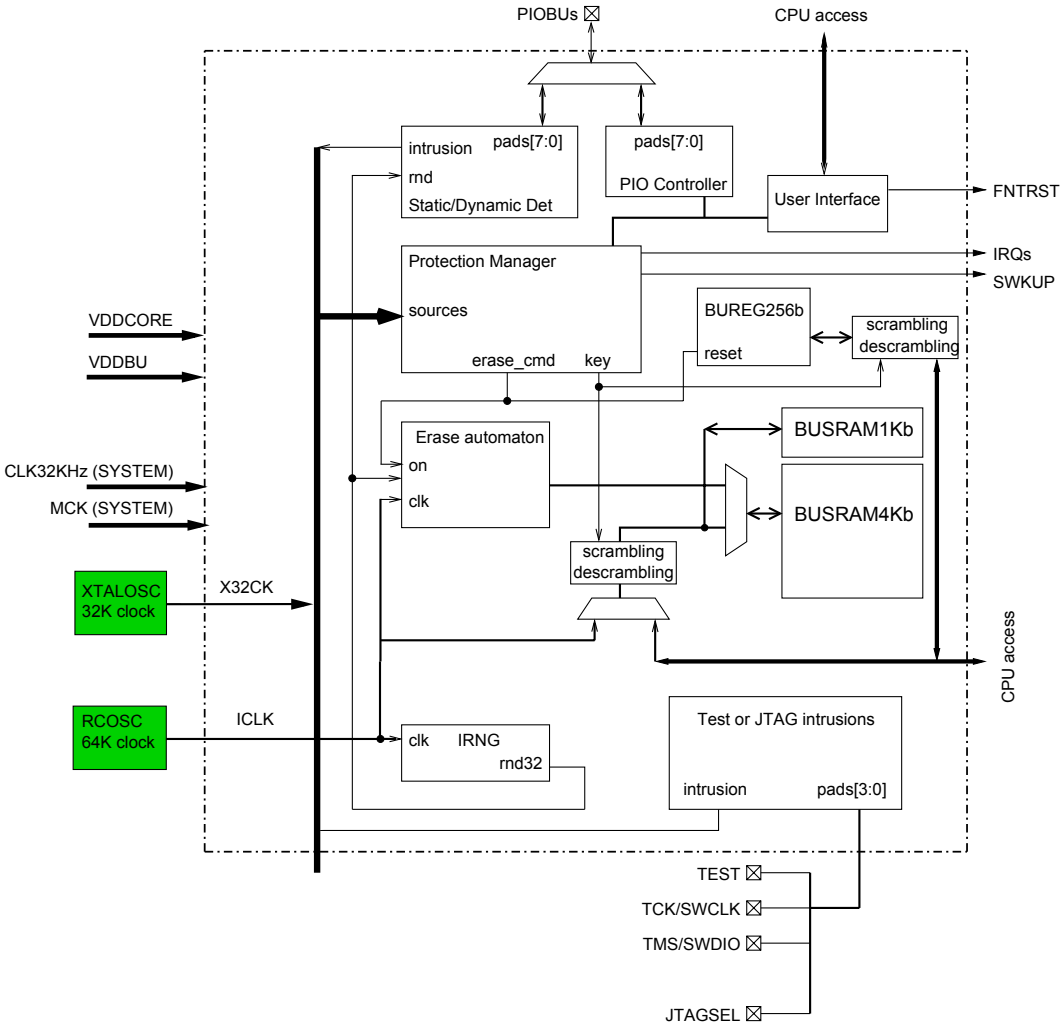
## 64.2 Embedded Characteristics

A PIO Controller managing up to eight pads (PIOBU) and offering:

- Standard I/O function powered in the backup domain
- Eight external switch state change detectors
- Memory erase and scrambling
- Backup SRAM access and Zeroisation process

## 64.3 Block Diagram

**Figure 64-1. SECUMOD Block Diagram**




The above figure represents the logic inside the SECUMOD. Analog cells are external to the IP and highlighted in green.

### 64.3.1 I/O Lines Description

**Table 64-1. I/O Lines Description**

| Name | Description | Type |
|---|---|---|
| CLK32KHZ | 32 kHz system clock from crystal or RC oscillator (SLCK) | Input |
| ICLK | 64 kHz RC Oscillator | Input |
| PIOBU[7:0] | Parallel IO backup controller, 8 pads | I/O |

| .....continued | | |
|---|---|---|
| **Name** | **Description** | **Type** |
| IRQ[1:0] | Interrupt signals going to secure AIC | Output |
| SWKUP | Wakeup signal going to system controller WKUP1 pin | Output |
| FNTRST | Force Cortex-A5 test port reset | Output |

## 64.4 Product Dependencies

### 64.4.1 Interrupt Sources

The SECUMOD provides two interrupt lines, each connected to one of the internal sources of the Advanced Interrupt Controller. Using these interrupts requires the AIC to be programmed first. Note that it is not recommended to use the interrupt lines in Edge-sensitive mode.

The first interrupt line (SECURAM ID) is dedicated to backup memories access right violations signaling, or end of erase (automatic or software erase) signaling.

The second interrupt line (SECUMOD ID) is shared by all the protection mechanisms.

See the Register Summary section for more information about interrupt acknowledgement.

The SECURAM and the SECUMOD interrupt lines are connected to the Interrupt Controller. The Interrupt Controller must be programmed before configuring the SECURAM or the SECUMOD.

## 64.5 Functional Description

### 64.5.1 Memory Mapping

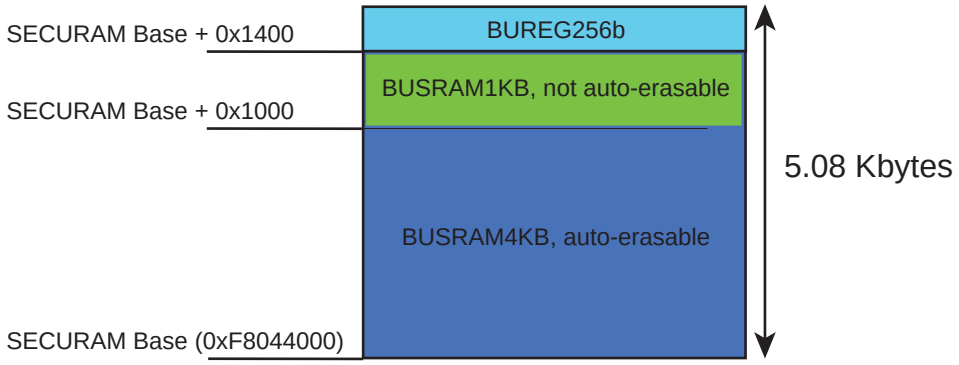
The SECUMOD embeds 5 Kbytes of SRAM split in two parts: the lower 4 Kbytes are erased in case of intrusion (BUSRAM4KB) while the upper 1 Kbyte is never erased (BUSRAM1KB). A 256-bit register bank is available as an additional memory and is totally erased in case of intrusion (BUREG256b).

All memories support 8-bit, 16-bit and 32-bit access sizes.

For power optimization, the transfers between the processor and these memories are decreased by a factor of 4.

The base address value of the SECURAM is 0xF8044000.

**Figure 64-2. SECUMOD Internal Memory Map**



### 64.5.2 Scrambling Keys

The secure memories (BUSRAM4KB, BUSRAM1KB and BUREG256b) are scrambled. The scrambling is enabled after reset and a scrambling key is automatically generated. The scrambling key can be modified through the Scrambling Key register (SECUMOD_SCRKEY).

Scrambling can be disabled using the Control register (SECUMOD_CR).

### 64.5.3 Internal Random Number Generator (IRNG)

The RNG cannot be read through the User Interface (a TrueRNG external to the SECUMOD is available for this purpose).

### 64.5.4 Protection Mechanisms

#### 64.5.4.1 PIO Backup Controller

The SECUMOD includes a PIO Controller powered by VDDBU which handles the eight PIOBU I/O pins.

Each I/O line is controlled by the PIO Controller and each pin can be configured to be driven. This is done by writing in the corresponding SECUMOD PIO Backup register (SECUMOD_PIOBUx). When SECUMOD_PIOBUx.OUTPUT is at ‘0’, the corresponding I/O line is used as an input only. When this bit is at ‘1’, the corresponding I/O line is driven by the PIO Backup Controller.

##### 64.5.4.1.1 Output Mode

When SECUMOD_PIOBUx.OUTPUT is set, the level driven on an I/O line can be determined by setting or clearing the PIO_SOD bit (Set Output Data). The value of this bit represents the data driven on the corresponding I/O line.

##### 64.5.4.1.2 Input Mode

The level on an I/O line can be read through the PIO_PDS bit (Pin Data Status) in the corresponding SECUMOD_PIOBUx. This bit indicates the level of the I/O line regardless of its configuration, whether as an input or driven by the PIO Controller.

##### 64.5.4.1.3 Static Intrusion Detectors and Programmable Internal Pullup/Pulldown

Intrusion detectors can be placed around the system to detect any intrusion attempt. This requires the corresponding I/O lines to be configured as inputs (SECUMOD_PIOBUx.OUTPUT = 0).

##### 64.5.4.1.4 Static Intrusion Detection

The detectors can be configured to detect either the rising edge or the falling edge on switches via SECUMOD_PIOBUx.SWITCH.

Example: A detector can consist of a normally-closed switch which sends a zero signal to the Protection Unit. When an intrusion attempt occurs, the switch state changes to an open position. The debounce filter waits until an intrusion has been detected for a programmable continuous period to send an alarm signal to the Protection Unit. This is to prevent erroneous intrusion detections.

##### 64.5.4.1.5 Internal Pullup/Pulldown

The user has the possibility to connect an internal pullup or pulldown (around 100 kΩ) by configuring SECUMOD_PIOBUx.PULLUP accordingly.

Configuring this field with a pullup or pulldown value activates the corresponding pullup/pulldown permanently.

**Note:** Internal pullups are connected at reset state.

##### 64.5.4.1.6 Scheduled Pullup/Pulldown

In order to reduce the power consumption on the VDDBU power supply, all activated pullups/pulldowns can be scheduled by following the steps below:

1. Activate the required pullup/pulldown.
2. Measure the level on the PIOBUx pin.
3. Deactivate the pullup/pulldown.

Scheduling is enabled by setting SECUMOD_PIOBUx.SCHEDULE.

**Note:** This feature is only effective if the PULLUP field indicates that a pullup or a pulldown is connected.

#### 64.5.4.1.7 Debouncing Time

The debouncing time is common to all I/Os. The principle is presented in the following figure. A period ($f_{ICLK}/2$) is allocated to each I/O. During that period, if SECUMOD_PIOBUx.SCHEDULE is set and if a pullup/pulldown is needed (PULLUP field different from 0), the pullup/pulldown is activated, the level is measured and the pullup/pulldown is deactivated. Otherwise, only the level is measured. Measurement is performed at the end of the allocated period.

**Figure 64-3. Schedule Principle**

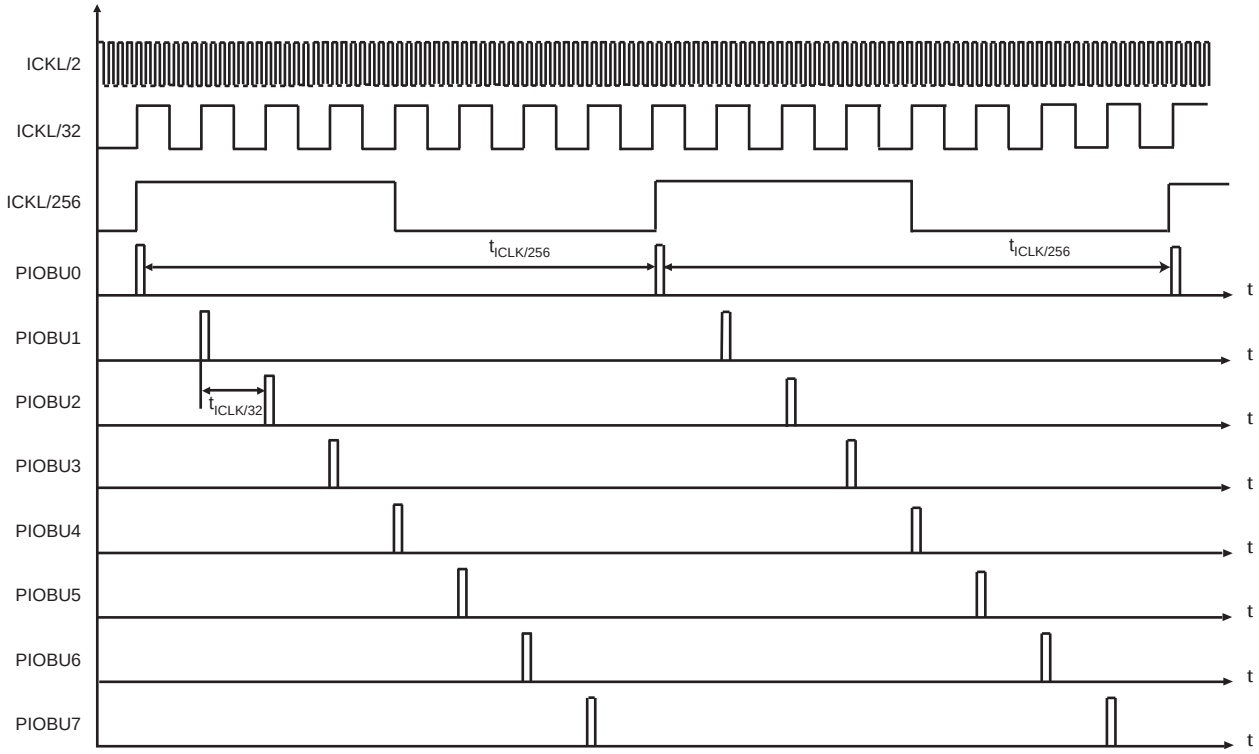



**Table 64-2. Timings vs. $f_{ICLK}$**

| Timing | $f_{ICLK}$ (kHz) | | | Units |
|---|---|---|---|---|
| | Min = 38 | Typ = 64 | Max = 90 | |
| $t_{ICLK/2}$ | 53 | 31 | 22 | µs |
| $t_{ICLK/32}$ | 842 | 500 | 356 | µs |
| $t_{ICLK/256}$ | 6.74 | 4.00 | 2.84 | ms |

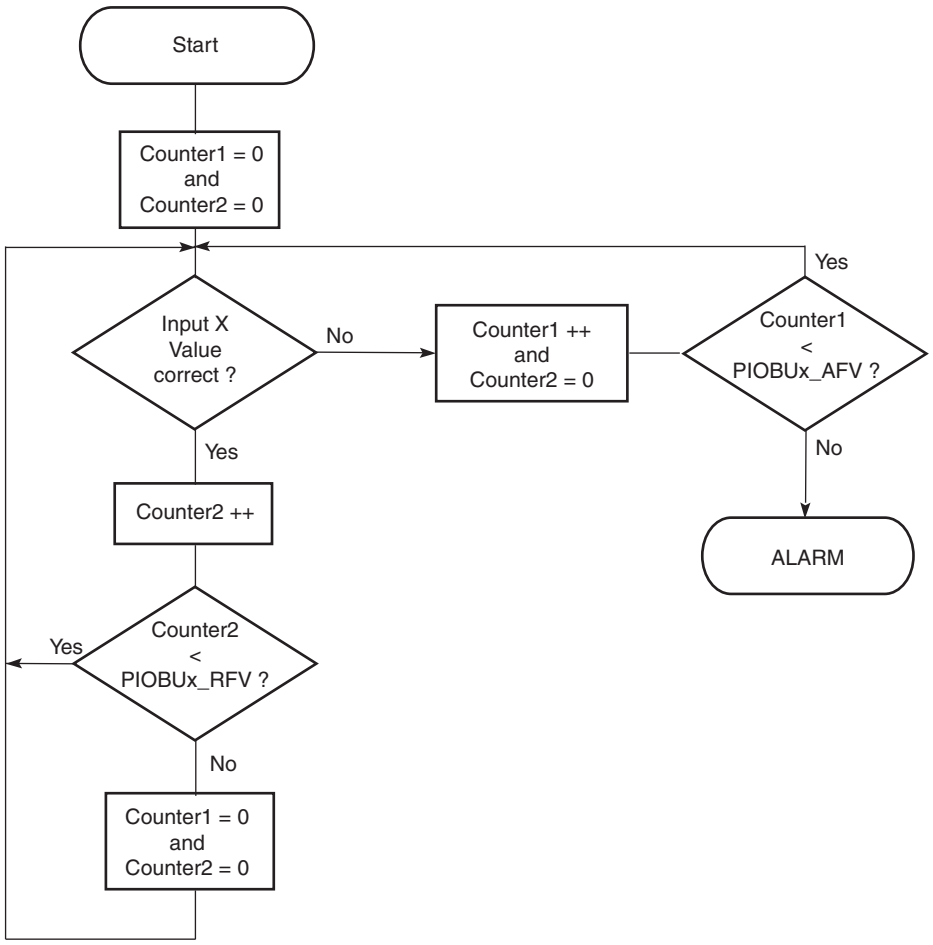
#### 64.5.4.1.8 PIOBUx Alarm Filtering in Static Mode

It is possible to filter the PIOBUx alarm detection by programming SECUMOD_PIOBUx.PIOBU_AFV. The steps are as follows:

1. A 9-bit counter is incremented each time the value present on the corresponding input is not the expected one.
2. An alarm is sent to the Protection Unit if the counter value reaches the value programmed in PIOBU_AFV.

The previous 9-bit counter is reset only if the value present on the input is correct and stable for a continuous programmable period defined by SECUMOD_PIOBUx.PIOBU_RFV (a second counter is used for that operation). See the figure below.

**Figure 64-4. PIOBUx Alarm Filtering Principle**




At reset state, the debouncers are not activated (PIOBU_AFV and PIOBU_RFV fields set to 0), which implies that no alarm can be generated.

Once both the PIOBU_AFV and the PIOBU_RFV fields have been programmed, the corresponding protection is activated and a CLR signal is generated automatically when an intrusion is detected. It is possible to generate an interrupt (or a wakeup signal) instead of clearing the secure memories content. To do so, the user must disable the protection in the Normal Mode Protection register (SECUMOD_NMPR) and configure the Normal Interrupt Enable Protection register (SECUMOD_NIEPR).

**Note:** If the Normal Mode Protection/Backup Mode Protection registers are not hidden, their configuration has priority on the debouncer activation in the PIOBUx configuration registers, which means that CLR signal generation is enabled/disabled in those two registers. Setting the PIOBU_AFV and PIOBU_RFV fields configure the debouncer sensitivity and does not generate any clear signal when an intrusion is detected.

**Table 64-3. Debouncing Time vs. $f_{ICLK}$**

| Debouncing Time | $f_{ICLK}$ (kHz) | | | Unit |
|---|---|---|---|---|
| | Min = 38 | Typ = 64 | Max = 90 | |
| Min (PIOBU_AFV = 1) | 6.74 | 4.00 | 2.84 | ms |
| Max (PIOBU_AFV = 9) | 3.45 | 2.05 | 1.46 | s |

**Note:** At reset state, the PIOBU_AFV and PIOBU_RFV fields are set to 0.

#### 64.5.4.2 JTAG Prevention

##### 64.5.4.2.1 Debug Interface Access Prevention

The SECUMOD can be used to block access to the system through the ARM processor's Debug Access Port interface. This feature is implemented via SECUMOD_JTAGCR, which enables assertion of the nDBGRESET reset input of the debug interface. Writing a ‘1’ to SECUMOD_JTAGCR.FNTRST prevents any activity on the TAP (Test Access Port) controller.

On standard devices, FNTRST resets to ‘0’ and thus does not prevent debug access.

FNTRST also locks the boundary scan when set.

##### 64.5.4.2.2 Physical Restrictions for JTAG Debug Mode

Invasive and non-invasive debug modes are controlled by four input pins of the Debug Access Port: DBGEN, SPIDEN, NIDEN and SPNIDEN.

In order to restrict the debug to nonsecure software parts only, the SEC_DEBUG_DIS fuse has to be configured in the customer fuse matrix.

Programming this fuse prevents JTAG secure debug irreversibly, but does not lock non-secure debug.

##### 64.5.4.2.3 Software Restrictions for JTAG Debug Mode

Setting SECUMOD_JTAGCR.CA5_DEBUG_MODE sets the DBGEN, SPIDEN, NIDEN and SPNIDEN Cortex inputs to the appropriate level in order to allow different debug permission levels. See 64.6.7 SECUMOD_JTAGCR for more information.

##### 64.5.4.2.4 Software Prevention for JTAG Debug

It is possible to prevent JTAG Debug accesses by forcing the reset signal of Debug Access Port by software.

While the reset signal is maintained low, the JTAG Debug interface cannot be used. To maintain the Debug Access Port in reset state, set SECUMOD_JTAGCR.FNTRST (in this case, Boundary JTAG is also disabled).

The key used for the BUREG256b scrambler/descrambler is derived from the BUSRAM4KB key and thus benefits from the same protection.

### 64.5.5 Erasing Secure Memories

#### 64.5.5.1 BUSRAM4KB Erase Sequence

##### 64.5.5.1.1 Principle

The BUSRAM4KB Erase sequence is controlled by an automaton which is activated by the CLR signal.

The following table shows the time to perform a partial erase (one erased word out of eight on the entire BUSRAM4KB), and the time to perform a full erase.

**Table 64-4. Erase Time Evaluation vs. $f_{ICLK}$**

| Erase | $f_{ICLK}$ Frequency (kHz) | | | Unit |
|---|---|---|---|---|
| | Min = 38 | Typ = 64 | Max = 90 | |
| Partial Erase | 1.68 | 1.00 | 0.71 | ms |
| Full Erase (4 Kbytes) | 13.47 | 8.00 | 5.69 | |

During the Erase sequence, the upper 1 Kbyte of memory (BUSRAM1KB) is still accessible by the system.

The erase is a write of random values instead of erase to zero.

#### 64.5.5.2 BUREG256b Erase Sequence

In parallel to the BUSRAM4KB Erase, the BUREG256b register bank is erased.

BUREG256b is always reset after a VDDBU powerup.

These registers are seen as zero after reset or after an erase.

##### 64.5.5.2.1 During and After BUSRAM4KB and BUREG256b Erase Sequence

Some flags can be read to know the real-time erase state of the memories. On completion of the Erase sequence, the SECURAM ID interrupt line is asserted.

### 64.5.6 Operating Modes

The SECUMOD is supplied by the VDDBU power supply. It is not possible to program the SECUMOD if VDDBU is not present.

The SECUMOD macrocell is able to operate in two different modes:

- When all supplies are present and can be monitored, the SECUMOD can be switched to Normal mode.
- Otherwise, the SECUMOD must be in Backup mode.

**Note:** After a powerup reset, the SECUMOD is in Backup mode.

The mode is selected by setting either SECUMOD_CR.NORMAL or SECUMOD_CR.BACKUP.

**Note:** The user must set SECUMOD_CR.BACKUP to enter Backup mode prior to shutting off the VDDCORE power supplies.

In both modes, the user can enable or disable a protection by writing in the corresponding Mode Protection register. See 64.5.7 Activation or Deactivation of Protections for more information.

#### 64.5.6.1 Protection Unit

The Protection Unit is used to centralize all alarms coming from the different monitors. When an alarm is detected, the Protection Unit sends a Clear signal to the automaton, which starts the secure memories Erase sequence if the memory is not empty.

The Protection Unit can also send:

- an IRQ interrupt signal (only in Normal mode)
- an SWKUP wakeup signal (only in Backup mode).

When an interrupt or a wakeup signal is generated, it is up to the user to detect the source of the alarm and to act accordingly, i.e., to clear the secure memories content or not.

As soon as an alarm is detected, the corresponding bit is set in the Status register (SECUMOD_SR). The only way to clear this bit is to set it in the Status Clear register (SECUMOD_SCR).

**Note:** Once a status bit is raised, it should not be cleared before the next slow clock period. If a clear does occur, the status bit rises again and the same alarm will be seen twice. To prevent this, it is recommended to wait at least one slow clock period after reading the Status register before clearing the status bits.

If a Clear of the secure memories content has been performed by the automaton, an ERASE_DONE flag is set to indicate that the secure memories content is not valid anymore. While the secure memories are erased, write accesses have no effect and read accesses return a static and invalid value (except for BUSRAM1KB).

### 64.5.7 Activation or Deactivation of Protections

It is possible to activate or deactivate each protection separately by writing in the Normal and Backup Mode Protection registers. These registers are hidden and the only way to make them appear is to write SECUMOD_CR.KEY with the correct value. This command field acts on a toggle basis: writing the correct value makes the registers appear and disappear.

At reset state, all protections are activated except the sixteen corresponding to the intrusion detectors (need to program PIOBUx).

### 64.5.8 Powerup Reset

After a powerup reset, the SECUMOD is in Backup mode, but in an unpredictable state.

The Slow Clock oscillator takes about one second to startup. It is also possible that monitors send alarms to the Protection Unit. However, a Clear command can be performed because the secure memories content is empty.

Care must be taken when writing in BUSRAM4KB or BUREG256b after reset. The user must make sure that no Erase sequence is running, otherwise the write access to BUSRAM4KB or BUREG256b is aborted. It is recommended to wait for the system to be established before accessing BUSRAM4KB or BUREG256b. This can last for at least one or two seconds. The verification is performed by reading the Status register. If there is no error for a continuous period (one second, for example), the user can access BUSRAM4KB or BUREG256b. If at least one error is detected, the user has to wait first for the ERASE_DONE flag to rise, and then wait again for at least one slow clock period after reading the Status register before writing content in the Status Clear register. At this stage, all status bits should be cleared. The user must then ensure that no error is raised in the Status register during the next second, for example.

## 64.6 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | SECUMOD_CR | 7:0 | | | | | | SWPROT | NORMAL | BACKUP |
| | | 15:8 | | | | | | SCRAMB[1:0] | | |
| | | 23:16 | KEY[7:0] | | | | | | | |
| | | 31:24 | KEY[15:8] | | | | | | | |
| 0x04 | SECUMOD_SYSR | 7:0 | SCRAMB | AUTOBKP | | | SWKUP | BACKUP | ERASE_ON | ERASE_DONE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | SECUMOD_SR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| | | 31:24 | | | | | | | | |
| 0x0C ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | SECUMOD_SCR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| | | 31:24 | | | | | | | | |
| 0x14 | SECUMOD_RAMRDY | 7:0 | | | | | | | | READY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | SECUMOD_PIOBU0 | 7:0 | PIOBU_RFV[3:0] | | | | PIOBU_AFV[3:0] | | | |
| | | 15:8 | SWITCH | SCHEDULE | PULLUP[1:0] | | | PIO_PDS | PIO_SOD | OUTPUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | SECUMOD_PIOBU1 | 7:0 | PIOBU_RFV[3:0] | | | | PIOBU_AFV[3:0] | | | |
| | | 15:8 | SWITCH | SCHEDULE | PULLUP[1:0] | | | PIO_PDS | PIO_SOD | OUTPUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | SECUMOD_PIOBU2 | 7:0 | PIOBU_RFV[3:0] | | | | PIOBU_AFV[3:0] | | | |
| | | 15:8 | SWITCH | SCHEDULE | PULLUP[1:0] | | | PIO_PDS | PIO_SOD | OUTPUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 | SECUMOD_PIOBU3 | 7:0 | PIOBU_RFV[3:0] | | | | PIOBU_AFV[3:0] | | | |
| | | 15:8 | SWITCH | SCHEDULE | PULLUP[1:0] | | | PIO_PDS | PIO_SOD | OUTPUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x28 | SECUMOD_PIOBU4 | 7:0 | PIOBU_RFV[3:0] | | | | PIOBU_AFV[3:0] | | | |
| | | 15:8 | SWITCH | SCHEDULE | PULLUP[1:0] | | | PIO_PDS | PIO_SOD | OUTPUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x2C | SECUMOD_PIOBU5 | 7:0 | PIOBU_RFV[3:0] | | | | PIOBU_AFV[3:0] | | | |
| | | 15:8 | SWITCH | SCHEDULE | PULLUP[1:0] | | | PIO_PDS | PIO_SOD | OUTPUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x30 | SECUMOD_PIOBU6 | 7:0 | PIOBU_RFV[3:0] | | | | PIOBU_AFV[3:0] | | | |
| | | 15:8 | SWITCH | SCHEDULE | PULLUP[1:0] | | | PIO_PDS | PIO_SOD | OUTPUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x34 | SECUMOD_PIOBU7 | 7:0 | PIOBU_RFV[3:0] | | | | PIOBU_AFV[3:0] | | | |
| | | 15:8 | SWITCH | SCHEDULE | PULLUP[1:0] | | | PIO_PDS | PIO_SOD | OUTPUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x38 ... 0x67 | Reserved | | | | | | | | | |
| 0x68 | SECUMOD_JTAGCR | 7:0 | | | | WZO | CA5_DEBUG_MODE[2:0] | | | FNTRST |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x6C ... 0x6F | Reserved | | | | | | | | | |
| 0x70 | SECUMOD_SCRKEY | 7:0 | SCRKEY[7:0] | | | | | | | |
| | | 15:8 | SCRKEY[15:8] | | | | | | | |
| | | 23:16 | SCRKEY[23:16] | | | | | | | |
| | | 31:24 | SCRKEY[31:24] | | | | | | | |
| 0x74 | SECUMOD_RAMACC | 7:0 | RW3[1:0] | | RW2[1:0] | | RW1[1:0] | | RW0[1:0] | |
| | | 15:8 | | | | | RW5[1:0] | | RW4[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x78 | SECUMOD_RAMACCSR | 7:0 | RW3[1:0] | | RW2[1:0] | | RW1[1:0] | | RW0[1:0] | |
| | | 15:8 | | | | | RW5[1:0] | | RW4[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x7C | SECUMOD_BMPR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x80 | SECUMOD_NMPR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| | | 31:24 | | | | | | | | |
| 0x84 | SECUMOD_NIEPR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| | | 31:24 | | | | | | | | |
| 0x88 | SECUMOD_NIDPR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| | | 31:24 | | | | | | | | |
| 0x8C | SECUMOD_NIMPR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| | | 31:24 | | | | | | | | |
| 0x90 | SECUMOD_WKPR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| | | 31:24 | | | | | | | | |

### 64.6.1 SECUMOD Control Register

**Name:** SECUMOD_CR
**Offset:** 0x0000
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SCRAMB[1:0] | | |
| Access | | | | | | W | W | |
| Reset | | | | | | – | – | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SWPROT | NORMAL | BACKUP |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

**Bits 31:16 – KEY[15:0]** Password
This command field acts on a toggle basis: writing the value 0x89CA alternatively makes the Normal or Backup Protection Registers appear and disappear.

Writing any other value in this field has no effect.

**Bits 10:9 – SCRAMB[1:0]** Memory Scrambling Enable

| Value | Description |
|---|---|
| 10 | Memories are not scrambled. |
| 01 | Memories are scrambled (default). |
| 00 | No effect |
| 11 | No effect |

**Bit 2 – SWPROT** Software Protection

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Starts the BUSRAM4KB and BUREG256b Clear content. |

**Bit 1 – NORMAL** Normal Mode

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Switches to Normal mode. |

**Bit 0 – BACKUP** Backup Mode

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Switches to Backup mode. |

### 64.6.2 SECUMOD System Status Register

**Name:** SECUMOD_SYSR
**Offset:** 0x0004
**Reset:** 0x000000D4
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SCRAMB | AUTOBKP | | | SWKUP | BACKUP | ERASE_ON | ERASE_DONE |
| Access | R | R | | | R | R | R | R/W |
| Reset | 1 | 1 | | | 0 | 1 | 0 | 0 |

**Bit 7 – SCRAMB** Scrambling Enabled

| Value | Description |
|---|---|
| 0 | Disabled. |
| 1 | Enabled. |

**Bit 6 – AUTOBKP** Automatic Backup Mode Enabled

| Value | Description |
|---|---|
| 0 | Disabled. |
| 1 | Enabled. |

**Bit 3 – SWKUP** SWKUP State

| Value | Description |
|---|---|
| 0 | No SWKUP signal sent since the last clear. |
| 1 | SWKUP signal has been sent since the last clear. |

**Bit 2 – BACKUP** Backup Mode

| Value | Description |
|---|---|
| 0 | Normal mode active. |
| 1 | Backup mode active. |

**Bit 1 – ERASE_ON** Erase Process Ongoing
When ERASE_ON returns to 0, ERASE_DONE is set after half a period of ICLK.

| ERASE_ON | ERASE_DONE | Status | Action |
|---|---|---|---|
| 0 | 0 | No Erase ongoing or since the last Erase. | Nothing. |
| 1 | 0 | An Erase process is running. | Wait until the ERASE_ON flag is reset.<br>ERASE_DONE will rise, see line below. |
| 0 | 1 | An Erase occurred and is finished. | Clear the ERASE_DONE flag. |
| 1 | 1 | An Erase process is running.<br>The ERASE_DONE flag refers to a previous Erase process, but was not cleared. | Wait until the ERASE_ON flag is reset, then clear the ERASE_DONE flag. |

| Value | Description |
|---|---|
| 0 | Erase automaton is not running. |
| 1 | Erase automaton is currently running, memories are not accessible. |

**Bit 0 – ERASE_DONE** Erasable Memories State

| Value | Description |
|---|---|
| 0 | Secure memories content has not been erased since the last clear. |
| 1 | Secure memories content has been erased since the last clear. The user must write 1 into this bit to clear this flag. Note that not clearing this flag does not prevent the next erase processes. This flag also activates the SECURAM interrupt line as long as it is not cleared. |

### 64.6.3 SECUMOD Status Register

**Name:** SECUMOD_SR
**Offset:** 0x0008
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: No alarm generated since the last clear.

1: An alarm has been generated by the corresponding monitor since the last clear.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – DETx** PIOBU Intrusion Detector

### 64.6.4 SECUMOD Status Clear Register

**Name:** SECUMOD_SCR
**Offset:** 0x0010
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Clears the corresponding alarm flag bit.

If the corresponding alarm was programmed to generate a SWKUP signal, clearing the alarm also clears the SWKUP status bit in the SECUMOD Status register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

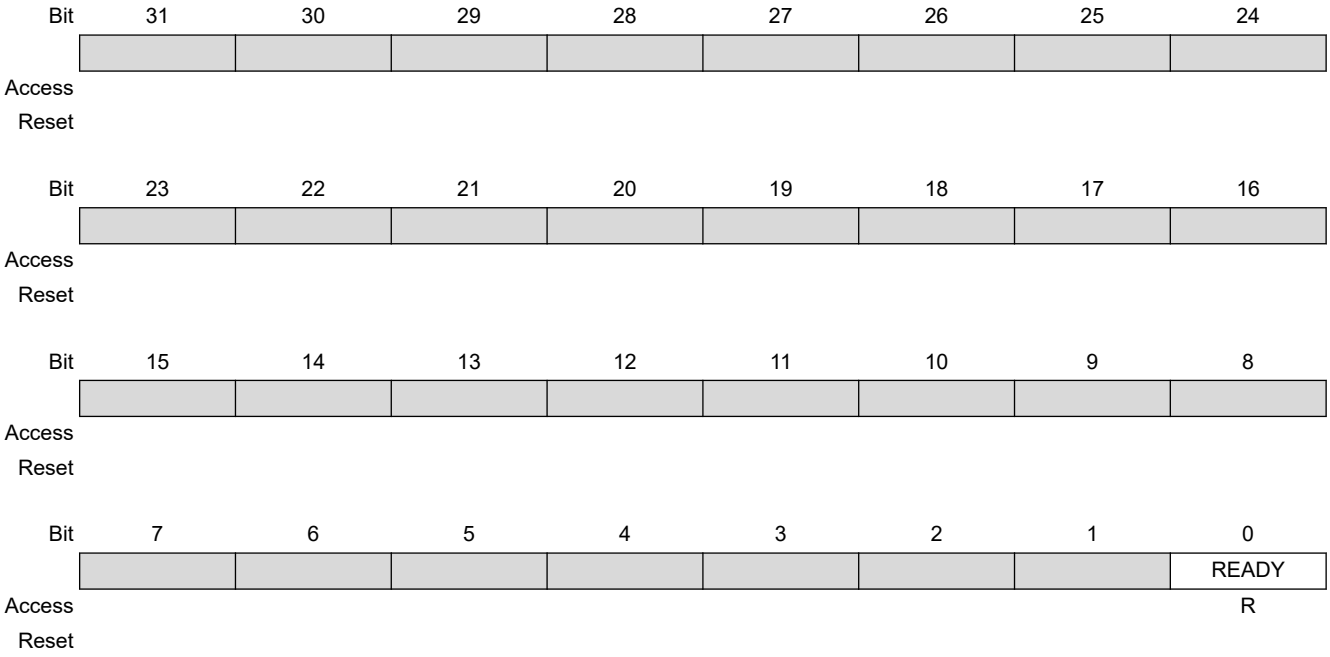
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – DETx** PIOBU Intrusion Detector

### 64.6.5 SECUMOD RAM Access Ready Register

**Name:** SECUMOD_RAMRDY
**Offset:** 0x0014
**Reset:** Undefined
**Property:** Read-only




**Bit 0 – READY** Ready for System Access Flag
When exiting Idle, System Reset or Backup mode, this flag must be read high before accessing the secure memories. The flag remains low until any ongoing process stops. Refer to the section “Real-time Clock (RTC) Register Summary” for more information.

### 64.6.6 SECUMOD PIO Backup Register x

**Name:** SECUMOD_PIOBUx
**Offset:** 0x18 + x*0x04 [x=0..7]
**Reset:** 0x00001000
**Property:** Read/Write

**Note:** The FILTER3_5 and DYNSTAT fields only exist for even PIOBUs.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SWITCH | SCHEDULE | PULLUP[1:0] | | | PIO_PDS | PIO_SOD | OUTPUT |
| Access | R/W | R/W | R/W | R/W | | R | R/W | R/W |
| Reset | 0 | 0 | 0 | 1 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PIOBU_RFV[3:0] | | | | PIOBU_AFV[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 15 – SWITCH** Switch State for Intrusion Detection

| Value | Description |
|---|---|
| 0 | Input default state is low level. |
| 1 | Input default state is high level. |

**Bit 14 – SCHEDULE** Pullup/Down Scheduled

| Value | Description |
|---|---|
| 0 | Pullup/pulldown is not scheduled. |
| 1 | Pullup/pulldown is scheduled. |

**Bits 13:12 – PULLUP[1:0]** Programmable Pullup State
This field is used to control the internal pullup or pulldown.

| PULLUP0 | | Status |
|---|---|---|
| 0 | 0 | No pullup/pulldown connected |
| 0 | 1 | Pullup connected |
| 1 | 0 | Pulldown connected |

| .....continued | | |
|---|---|---|
| **PULLUP0** | | **Status** |
| 1 | 1 | Reserved |

**Bit 10 – PIO_PDS** Level on the Pin in Input Mode (OUTPUT = 0)

| Value | Description |
|---|---|
| 0 | The I/O line is at level 0. |
| 1 | The I/O line is at level 1. |

**Bit 9 – PIO_SOD** Set/Clear the I/O Line when configured in Output Mode (OUTPUT =1)

| Value | Description |
|---|---|
| 0 | Clears the data to be driven on the I/O line. |
| 1 | Sets the data to be driven on the I/O line. |

**Bit 8 – OUTPUT** Configure I/O Line in Input/Output

| Value | Description |
|---|---|
| 0 | The I/O line is a pure input. |
| 1 | The I/O line is enabled in output. |

**Bits 7:4 – PIOBU_RFV[3:0]** PIOBUx Reset Filter Value
This field is used to define the number of consecutive valid states to be reached before resetting the AFV counter.

| PIOBU_RFV | Maximum Counter Value |
|---|---|
| 0 | 0 (No static protection) |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |
| 8 | 256 |
| 9 | 512 |

This field must be set to 0 when Dynamic Intrusion is selected.

**Bits 3:0 – PIOBU_AFV[3:0]** PIOBU Alarm Filter Value
This field is used to define the filter value prior to generating an alarm.

| PIOBU_AFV | Maximum Counter Value |
|---|---|
| 0 | 0 (No static protection) |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |
| 8 | 256 |
| 9 | 512 |

This field must be set to 0 when Dynamic Intrusion is selected.

### 64.6.7 SECUMOD JTAG Protection Control Register

**Name:** SECUMOD_JTAGCR
**Offset:** 0x0068
**Reset:** 0x00000008
**Property:** Read/Write

**Note:** Reset values are all 0 when fuse DEFDBG is programmed.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | WZO | CA5_DEBUG_MODE[2:0] | | | FNTRST |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 1 | 0 | 0 | 0 |

**Bit 4 – WZO** Write ZERO
Must be written with 0.

**Bits 3:1 – CA5_DEBUG_MODE[2:0]** Cortex-A5 Invasive/Non-Invasive Secure/Non-Secure Debug Permissions
This field is used to set different debug permission levels. For instance, it can be used to prevent debug on secure parts of the code. The table below shows the effect of the field value on the Cortex-A5 pins (SPIDEN, DBGEN, SPNIDEN and NIDEN).

| CA5_DEBUG_MODE Value | Cortex-A5 Debug Permissions | SPIDEN | DBGEN | SPNIDEN | NIDEN |
|---|---|---|---|---|---|
| b000 | No Debug | 0 | 0 | 0 | 0 |
| b001 | Non-Invasive, Non-Secure | 0 | 0 | 0 | 1 |
| b010 | Full Non-Secure (Invasive and Non-Invasive) | 0 | 1 | 0 | 0 |
| b011 | Full Non-Secure + Non-Invasive Secure | 0 | 1 | 1 | 1 |
| b100 | Full Debug allowed | 1 | 1 | 1 | 1 |

**Bit 0 – FNTRST** Force NTRST

| Value | Description |
|---|---|
| 0 | The ARM processor's TAP controller access and Boundary JTAG are not blocked by the SECUMOD. |
| 1 | nDBGRESET of the ARM processor's TAP controller and Boundary JTAG reset are held low, preventing the processor to switch to debug state and Boundary JTAG to work. |

### 64.6.8 SECUMOD Scrambling Key Register

**Name:** SECUMOD_SCRKEY
**Offset:** 0x0070
**Reset:** Undefined
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SCRKEY[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SCRKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SCRKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SCRKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | | | | | | | | |

**Bits 31:0 – SCRKEY[31:0]** Scrambling Key Value
This 32-bit key is used by the secure memories scrambler/descrambler logics. When changed, the readable content of the memories is made unintelligible instantaneously.

### 64.6.9 SECUMOD RAM Access Rights Register

**Name:** SECUMOD_RAMACC
**Offset:** 0x0074
**Reset:** 0x00003FFFF
**Property:** Read/Write

The following configuration values are valid for all listed bit names of this register:

00: No access allowed

01: Only write access allowed

10: Only read access allowed

11: Read and write accesses allowed

Accessing a forbidden area causes an interrupt (SECURAM ID).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RW5[1:0] | | RW4[1:0] | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RW3[1:0] | | RW2[1:0] | | RW1[1:0] | | RW0[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 11:10 – RW5[1:0]** Access right for RAM region [5 Kbytes; 6 Kbytes] (register bank BUREG256b)

**Bits 9:8 – RW4[1:0]** Access right for RAM region [4 Kbytes; 5 Kbytes]

**Bits 7:6 – RW3[1:0]** Access right for RAM region [3 Kbytes; 4 Kbytes]

**Bits 5:4 – RW2[1:0]** Access right for RAM region [2 Kbytes; 3 Kbytes]

**Bits 3:2 – RW1[1:0]** Access right for RAM region [1 Kbyte; 2 Kbytes]

**Bits 1:0 – RW0[1:0]** Access right for RAM region [0; 1 Kbyte]

### 64.6.10 SECUMOD RAM Access Rights Status Register

**Name:** SECUMOD_RAMACCSR
**Offset:** 0x0078
**Reset:** 0x00000000
**Property:** Read/Write

The following configuration values are valid for all listed bit names of this register:

00: No access violation occurred

01: Write access violation occurred

10: Read access violation occurred

11: Read and write access violation occurred

Writing any value to this register resets the register and the associated interrupt line (SECURAM ID).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RW5[1:0] | | RW4[1:0] | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RW3[1:0] | | RW2[1:0] | | RW1[1:0] | | RW0[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 11:10 – RW5[1:0]** Access right status for RAM region [5 Kbytes; 6 Kbytes] (register bank BUREG256b)

**Bits 9:8 – RW4[1:0]** Access right status for RAM region [4 Kbytes; 5 Kbytes]

**Bits 7:6 – RW3[1:0]** Access right status for RAM region [3 Kbytes; 4 Kbytes]

**Bits 5:4 – RW2[1:0]** Access right status for RAM region [2 Kbytes; 3 Kbytes]

**Bits 3:2 – RW1[1:0]** Access right status for RAM region [1 Kbytes; 2 Kbytes]

**Bits 1:0 – RW0[1:0]** Access right status for RAM region [0; 1 Kbyte]

### 64.6.11 SECUMOD Backup Mode Protection Register

**Name:** SECUMOD_BMPR
**Offset:** 0x007C
**Reset:** 0xFFFF0CCF
**Property:** Read/Write

PIO backup protections are off after backup reset whatever the reset value of this register. See SECUMOD_PIOBUx register descriptions to enable these protections.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – DETx** PIOBU Intrusion Detector Protection
Reminder: Enabling PIOBU protection requires additional programming of PIOBUx registers.

| Value | Description |
|---|---|
| 0 | Protection disabled. |
| 1 | Protection enabled. |

### 64.6.12 SECUMOD Normal Mode Protection Register

**Name:** SECUMOD_NMPR
**Offset:** 0x0080
**Reset:** 0xFFFFFFFF
**Property:** Read/Write

PIO backup protections are off after backup reset whatever the reset value of this register. See SECUMOD_PIOBUx register descriptions to enable these protections.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – DETx** PIOBU Intrusion Detector Protection
Reminder: Enabling PIOBU protection requires additional programming of PIOBUx registers.

| Value | Description |
|---|---|
| 0 | Protection disabled. |
| 1 | Protection enabled. |

### 64.6.13 SECUMOD Normal Interrupt Enable Protection Register

**Name:** SECUMOD_NIEPR
**Offset:** 0x0084
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – DETx** PIOBU Intrusion Detector Protection Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding interrupt. |

### 64.6.14 SECUMOD Normal Interrupt Disable Protection Register

**Name:** SECUMOD_NIDPR
**Offset:** 0x0088
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – DETx** PIOBU Intrusion Detector Protection Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding interrupt. |

### 64.6.15 SECUMOD Normal Interrupt Mask Protection Register

**Name:** SECUMOD_NIMPR
**Offset:** 0x008C
**Reset:** 0x00000000
**Property:** Read-only

The reset values apply after Peripheral Reset (other reset values are defined after Backup Reset).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – DETx** PIOBU Intrusion Detector Protection Interrupt Mask

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is disabled. |
| 1 | The corresponding interrupt is enabled. |

### 64.6.16 SECUMOD Wakeup Register

**Name:** SECUMOD_WKPR
**Offset:** 0x0090
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DET7 | DET6 | DET5 | DET4 | DET3 | DET2 | DET1 | DET0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – DETx** PIOBU Intrusion Detector Protection

| Value | Description |
|---|---|
| 0 | No wakeup signal is generated if the corresponding alarm is detected. |
| 1 | A wakeup signal (SWKUP) is generated if the corresponding alarm is detected. |

# 65. Analog-to-Digital Controller (ADC)

## 65.1 Description

The ADC is based on a 12-bit Analog-to-Digital Converter (ADC) managed by an ADC Controller providing enhanced resolution up to 14 bits. See 65.3 Block Diagram. It also integrates a 12-to-1 analog multiplexer, making possible the analog-to-digital conversions of 12 analog lines. The conversions extend from 0V to the voltage carried on pin ADVREF.

Conversion results are reported in a common register for all channels, as well as in a channel-dedicated register.

The 13-bit and 14-bit resolution modes are obtained by averaging multiple samples to decrease quantization noise. For the 13-bit mode, 4 samples are used, which gives a real sample rate of 1/4 of the actual sample frequency. For the 14-bit mode, 16 samples are used, giving a real sample rate of 1/16 of the actual sample frequency. This arrangement allows conversion speed to be traded off against for better accuracy.

The software trigger, external trigger on rising edge of the ADTRG pin or internal triggers from Timer Counter output(s) are configurable.

The comparison circuitry allows automatic detection of values below a threshold, higher than a threshold, in a given range or outside the range, thresholds and ranges being fully configurable.

The ADC Controller internal fault output is directly connected to the PWM fault input. This input can be asserted by means of comparison circuitry to immediately put the PWM output in a safe state (pure combinational path).

The ADC also integrates a Sleep mode and a conversion sequencer and connects with a DMA channel. These features reduce both power consumption and processor intervention.

This ADC has a selectable single-ended or fully differential input.

This ADC Controller includes a Resistive Touchscreen Controller. It supports 4-wire and 5-wire technologies.

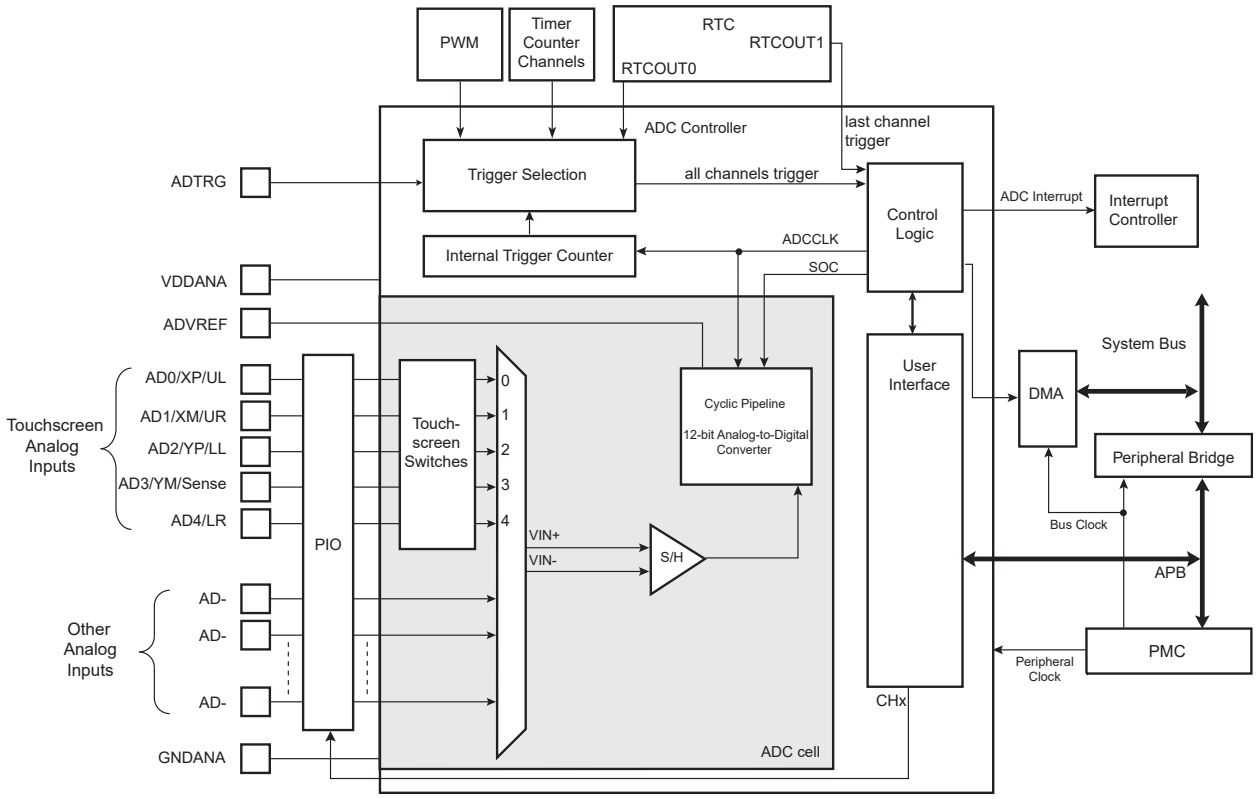
## 65.2 Embedded Characteristics

- 12-bit Resolution with Enhanced Mode up to 14 bits
- 1 Msps Conversion Rate
- Digital Averaging Function providing Enhanced Resolution Mode up to 14 bits
- Wide Range of Power Supply Operation
- Selectable Single-Ended or Differential Input Voltage
- Digital correction of offset and gain errors
- Resistive 4-wire and 5-wire Touchscreen Controller
  - Position and Pressure Measurement for 4-wire Screens
  - Position Measurement for 5-wire Screens
  - Average of Up to 8 Measures for Noise Filtering
- Programmable Pen Detection Sensitivity
- Integrated Multiplexer Offering Up to 12 Independent Analog Inputs

- Individual Enable and Disable of Each Channel
- Hardware or Software Trigger from:
  - External Trigger Pin
  - Timer Counter Outputs (Corresponding TIOA Trigger)
  - ADC Internal Trigger Counter
  - Trigger on Pen Contact Detection
  - PWM Event Line
- Drive of PWM Fault Input
- DMA Support
- Two Sleep Modes (Automatic Wakeup on Trigger)
  - Lowest Power Consumption (Voltage Reference OFF Between Conversions)
  - Fast Wakeup Time Response on Trigger Event (Voltage Reference ON Between Conversions)
- Channel Sequence Customizing
- Automatic Window Comparison of Converted Values
- Asynchronous Partial Wakeup (SleepWalking) on External Trigger
- Register Write Protection

## 65.3 Block Diagram

**Figure 65-1. Analog-to-Digital Converter Block Diagram**



## 65.4 Signal Description

**Table 65-1. ADC Pin Description**

| Pin Name | Description |
|---|---|
| VDDANA | Analog power supply |
| ADVREF | Reference voltage |
| AD0–AD11 | Analog input channels |
| ADTRG | External trigger |

## 65.5 Product Dependencies

### 65.5.1 Power Management

The ADC Controller is not continuously clocked. The programmer must first enable the ADC Controller peripheral clock in the Power Management Controller (PMC) before using the ADC Controller. However, if the application does not require ADC operations, the ADC Controller clock can be stopped when not needed and restarted when necessary. Configuring the ADC Controller does not require the ADC Controller clock to be enabled.

### 65.5.2 Interrupt Sources

The ADC interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the ADC interrupt requires the interrupt controller to be programmed first.

### 65.5.3 I/O Lines

The digital input ADTRG is multiplexed with digital functions on the I/O line and the selection of ADTRG is made using the PIO controller.

The ADC_ADx analog inputs are multiplexed with digital functions on the I/O lines. ADC_ADx inputs are selected as inputs of the ADCC when writing a one in the corresponding ADC_CHER.CHx bit and the digital functions are not selected.

### 65.5.4 Hardware Triggers

The ADC can use internal signals to start conversions. See the ADC_MR.TRGSEL field description in 65.7.2 ADC_MR for exact wiring of internal triggers.

### 65.5.5 Fault Output

The ADC Controller has the FAULT output connected to the FAULT input of PWM. See 65.6.18 Fault Event and section “Pulse Width Modulation Controller (PWM)”.

## 65.6 Functional Description

### 65.6.1 Analog-to-Digital Conversion

Once the programmed startup time (ADC_MR.STARTUP) has elapsed, ADC conversions are sequenced by three operating times:

- Tracking time—the time for the ADC to charge its input sampling capacitor to the input voltage. When several channels are converted consecutively, the inherent tracking time is 6 ADC clock

cycles. However, the tracking time can be increased using the TRACKTIM field in the Mode register (ADC_MR).

- ADC inherent conversion time—the time for the ADC to convert the sampled analog voltage. This time is constant and is defined from start of conversion to end of conversion.
- Channel conversion period—the effective time between the end of the current channel conversion and the end of the next channel conversion.

**Figure 65-2. Sequence of Consecutive ADC Conversions with TRACKTIM = 0**

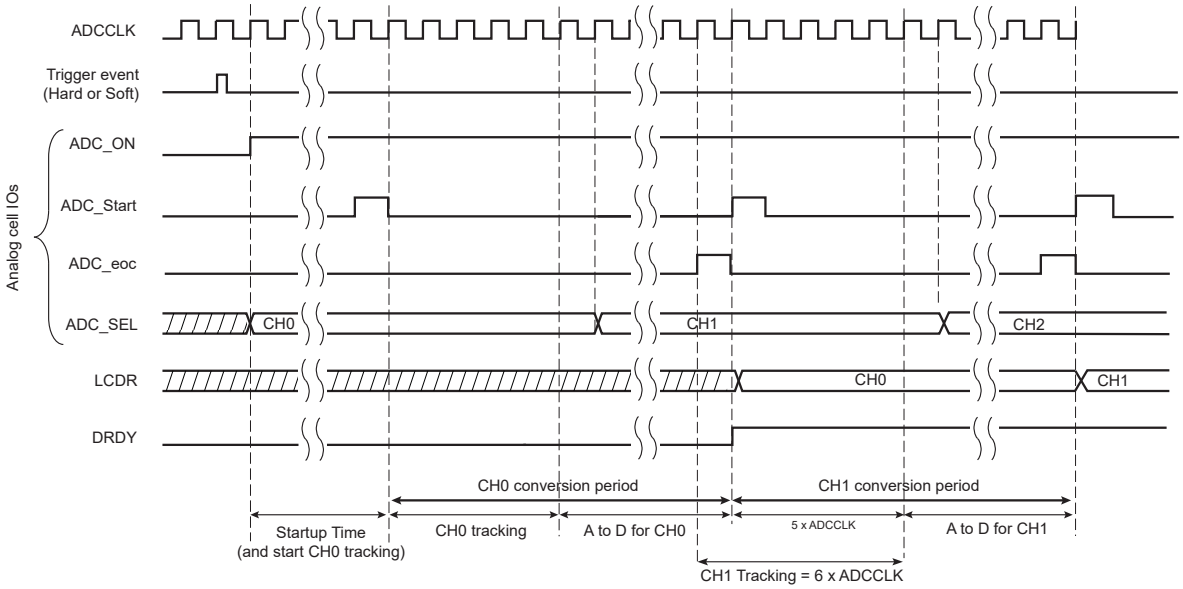
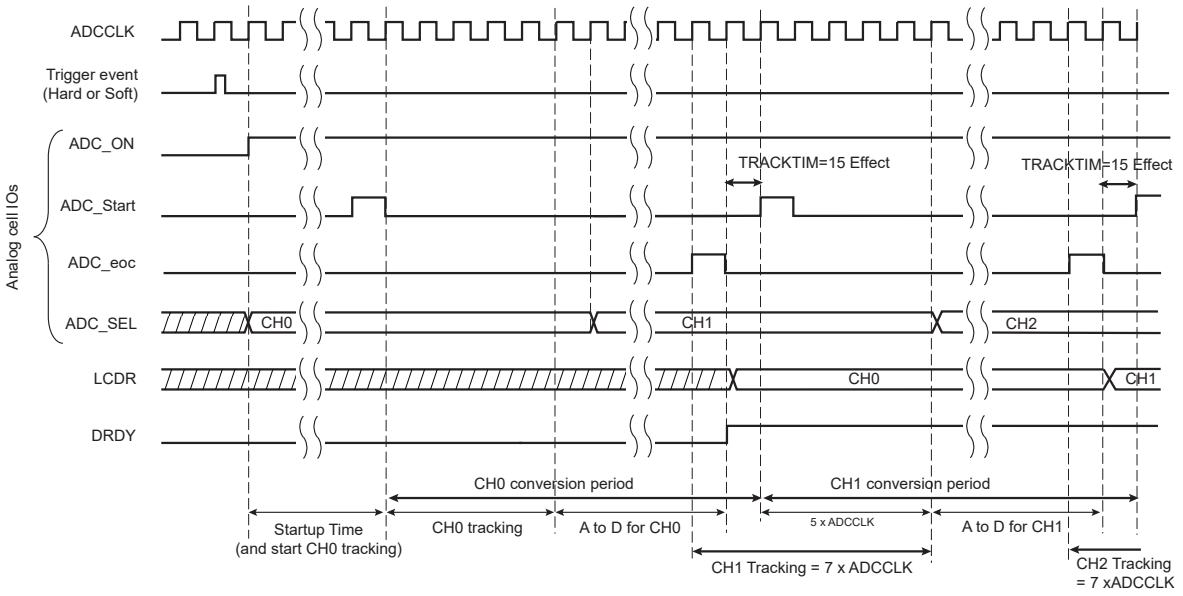



**Figure 65-3. Sequence of Consecutive ADC Conversions with TRACKTIM = 15**




### 65.6.2 ADC Clock

The ADC uses the ADC clock (ADCCLK) to perform conversions. The ADC clock frequency is selected in the ADC_MR.PRESCAL.

To generate the ADC clock, the prescaler has two clock sources: the peripheral clock and the GCLK clock. This clock source is selected using the SRCCLK bit in the Extended Mode register (ADC_EMR).

If GCLK is selected as a source clock, the ADC clock frequency is independent of the processor/bus clock. At reset, the peripheral clock is selected.

If the ADC_EMR.SRCCLK bit is cleared, the prescaler clock (presc_clk) is driven by peripheral_clock. If the ADC_EMR.SRCCLK bit is set, the prescaler clock is driven by GCLK. The ADC clock frequency is between fpresc_clk/2, if PRESCAL is 0, and fpresc_clk/512, if PRESCAL is set to 255 (0xFF).

PRESCAL must be programmed to provide the ADC clock frequency parameter provided in the “Electrical Characteristics” section.

### 65.6.3 ADC Reference Voltage

The voltage reference input of the ADC is the ADVREF pin. Refer to the “Electrical Characteristics” section for further details.

### 65.6.4 Conversion Resolution

The ADC has a native resolution of 12 bits.

The ADC Controller provides enhanced resolution up to 14 bits by means of digital averaging.

If ADTRG is asynchronous to the ADC peripheral clock, the internal resynchronization introduces a jitter of 1 peripheral clock. This jitter may reduce the resolution of the converted signal.

The same applies when using the independent clock (ADC_MR.SRCCLK = 1), if the provided clock is asynchronous to ADC peripheral clock.

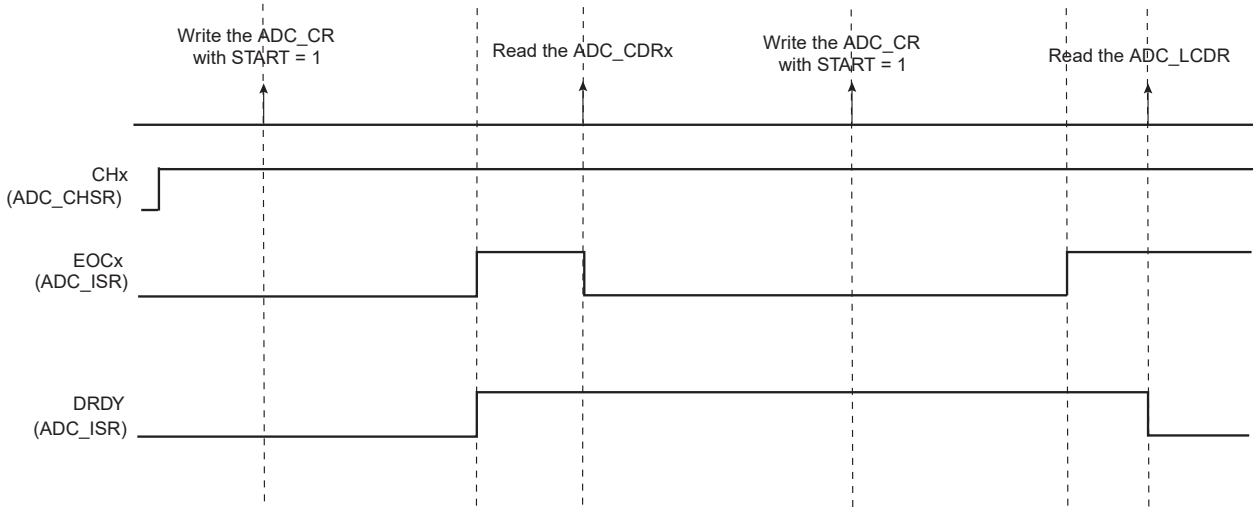
### 65.6.5 Conversion Results

When a conversion is completed, the resulting digital value is stored in the Channel Data register (ADC_CDRx) of the current channel and in the Last Converted Data register (ADC_LCDR). By setting the TAG option in ADC_EMR, ADC_LCDR presents the channel number associated with the last converted data in the CHNB field.

When a conversion is completed, the channel EOC bit and the DRDY bit in the Interrupt Status register (ADC_ISR) are set. In the case of a connected DMA channel, DRDY rising triggers a data request. In any case, either EOC and DRDY can trigger an interrupt.

Reading one of the ADC_CDRx clears the corresponding EOC bit. Reading ADC_LCDR clears the DRDY bit.

**Figure 65-4. EOCx and DRDY Flag Behavior**

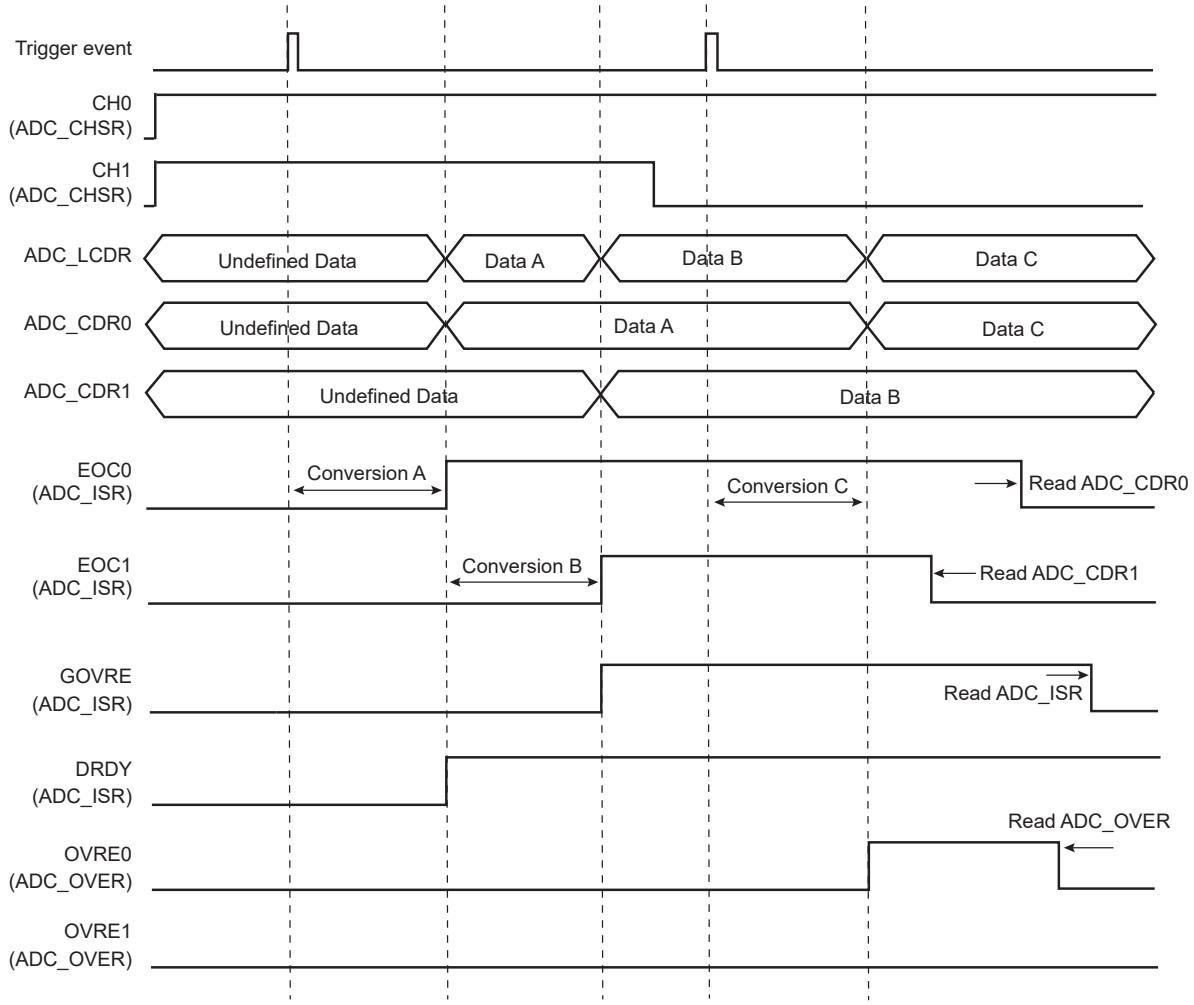


If ADC_CDR is not read before further incoming data is converted, the corresponding OVREx flag is set in the Overrun Status register (ADC_OVER).

If new data is converted when DRDY is high, the ADC_ISR.GOVRE bit is set.

The OVREx flag is automatically cleared when ADC_OVER is read, and the GOVRE flag is automatically cleared when ADC_ISR is read.

**Figure 65-5. EOCx, OVREx and GOVREx Flag Behavior**




**WARNING** If the corresponding channel is disabled during a conversion or if it is disabled and then reenabled during a conversion, its associated data and corresponding EOCx and GOVRE flags in ADC_ISR and OVREx flags in ADC_OVER are unpredictable.

### 65.6.6 Conversion Results Format

The conversion results can be signed (2's complement) or unsigned depending on the value of the ADC_EMR.SIGNMODE field.

If conversion results are signed and resolution is less than 16 bits, the sign is extended up to the bit 15 (e.g., 0xF43 for 12-bit resolution is read as 0xFF43, and 0x467 is read as 0x0467).

### 65.6.7 Conversion Triggers

Conversions of the active analog channels are started with a software or hardware trigger. The software trigger is provided by writing the Control register (ADC_CR) with the START bit at 1.

The list of external/internal events is provided in 65.7.2 ADC_MR. The hardware trigger is selected using the ADC_MR.TRGSEL field. The selected hardware trigger is enabled if TRGMOD = 1, 2 or 3 in the Trigger register (ADC_TRGR).

The ADC also provides a dual trigger mode (ADC_LCTMR.DUALTRIG = 1) in which the higher index channel can be sampled at a rhythm different from the other channels. The trigger of the last channel is generated by the RTC. See 65.6.12 Last Channel Specific Measurement Trigger.

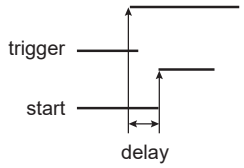
The ADC_TRGR.TRGMOD field selects the hardware trigger from the following:

- any edge, either rising or falling or both, detected on the external trigger pin ADTRG
- the Pen Detect, depending on how the PENDET bit is set in the Touchscreen Mode register (ADC_TSMR)
- a continuous trigger, meaning the ADC Controller restarts the next sequence as soon as it finishes the current one
- a periodic trigger, which is defined by programming the ADC_TRGR.TRGPER field

The minimum time between two consecutive trigger events must be strictly greater than the duration time of the longest conversion sequence according to configuration of registers ADC_MR, ADC_CHSR, ADC_SEQRx, and ADC_TSMR.

If a hardware trigger is selected, the start of a conversion is triggered after a delay starting at each rising edge of the selected signal. Due to asynchronous handling, the delay may vary in a range of two peripheral clock periods to one ADC clock period. This delay introduces sampling jitter in the A/D conversion process and may therefore degrade the conversion performance (e.g., SNR, THD).

**Figure 65-6. Hardware Trigger Delay**




If one of the TIOA outputs is selected, the corresponding Timer Counter channel must be programmed in Waveform mode.

Only one start command is necessary to initiate a conversion sequence on all the channels. The ADC hardware logic automatically performs the conversions on the active channels, then waits for a new request. The Channel Enable (ADC_CHER) and Channel Disable (ADC_CHDR) registers enable the analog channels to be enabled or disabled independently.

If the ADC is used with a DMA, only the transfers of converted data from enabled channels are performed and the resulting data buffers should be interpreted accordingly.

### 65.6.8 Sleep Mode and Conversion Sequencer

The ADC Sleep mode maximizes power saving by automatically deactivating the ADC when it is not being used for conversions. Sleep mode is selected by setting the ADC_MR.SLEEP bit.

Sleep mode is managed by a conversion sequencer, which automatically processes the conversions of all channels at lowest power consumption.

This mode can be used when the minimum period of time between two successive trigger events is greater than the startup period of the ADC. Refer to section “Electrical Characteristics”.

When a start conversion request occurs, the ADC is automatically activated. As the analog cell requires a startup time, the logic waits during this time and starts the conversion on the enabled channels. When all conversions are complete, the ADC is deactivated until the next trigger. Events triggered during the sequence are ignored.

The conversion sequencer allows automatic processing with minimum processor intervention and optimized power consumption. Conversion sequences can be performed periodically using the internal timer (ADC_TRGR) or the PWM event line. The periodic acquisition of several samples can be processed automatically without any intervention of the processor via the DMA.

The sequence can be customized by programming the Sequence Channel registers ADC_SEQR1 and ADC_SEQR2 and setting the USEQ bit of the Mode register (ADC_MR). The user can choose a specific order of channels and can program up to 12 conversions by sequence. The user is free to create a personal sequence by writing channel numbers in ADC_SEQR1 and ADC_SEQR2. Not only can channel numbers be written in any sequence, channel numbers can be repeated several times. When the ADC_MR.USEQ bit is set, the ADC_SEQR1.USCHx and ADC_SEQR2.USCHx fields are used to define the sequence. Only enabled USCHx fields will be part of the sequence. Each USCHx field has a corresponding enable, CHx-1, in ADC_CHER.

If all ADC channels (i.e., 12) are used on an application board, there is no restriction of usage of the user sequence. However, if some ADC channels are not enabled for conversion but rather used as pure digital inputs, the respective indexes of these channels cannot be used in the user sequence fields (see ADC_SEQRx). For example, if channel 4 is disabled (ADC_CHSR[4] = 0), ADC_SEQRx fields USCH1 up to USCH12 must not contain the value 4. Thus the length of the user sequence may be limited by this behavior.

As an example, if only four channels over 12 (CH0 up to CH3) are selected for ADC conversions, the user sequence length cannot exceed four channels. Each trigger event may launch up to four successive conversions of any combination of channels 0 up to 3 but no more (i.e., in this case the sequence CH0, CH0, CH1, CH1, CH1 is impossible).

A sequence that repeats the same channel several times requires more enabled channels than channels actually used for conversion. For example, the sequence CH0, CH0, CH1, CH1 requires four enabled channels (four free channels on application boards) whereas only CH0, CH1 are really converted.

**Note:** The reference voltage pins always remain connected in Normal mode as in Sleep mode.

### 65.6.9 Comparison Window

The ADC Controller features automatic comparison functions. It compares converted values to a low threshold, a high threshold or both, depending on the value of the ADC_EMR.CMPMODE field. The comparison can be done on all channels or only on the channel specified in the ADC_EMR.CMPSEL field. To compare all channels, the ADC_EMR.CMPALL bit must be set.

If set, the ADC_EMR.CMPTYPE bit can be used to discard all conversion results that do not match the comparison conditions. Once a conversion result matches the comparison conditions, all the subsequent conversion results are stored in ADC_LCDR (even if these results do not meet the comparison conditions). Setting the ADC_CR.CMPRST bit immediately stops the conversion result storage until the next comparison match.

If the ADC_EMR.CMPTYPE bit is cleared, all conversions are stored in ADC_LCDR. Only the conversions that match the comparison conditions trigger the ADC_ISR.COMPE flag.

Moreover, a filtering option can be set by writing the number of consecutive comparison matches needed to raise the flag. This number can be written and read in the ADC_EMR.CMPFILTER field. The filtering option is dedicated to reinforcing the detection of an analog signal overpassing a predefined threshold. The filter is cleared as soon as ADC_ISR is read, so this filtering function must be used with peripheral DMA controller and works only when using Interrupt mode (no polling).

The flag can be read on the ADC_ISR.COMPE bit and can trigger an interrupt.

The high threshold and the low threshold can be read/write in the Compare Window register (ADC_CWR).

Depending on the sign of the conversion, chosen with the ADC_EMR.SIGNMODE field, the high threshold and low threshold values must be signed or unsigned to maintain consistency during the comparison. If the conversion is signed, both thresholds must also be signed; if the conversion is unsigned, both thresholds must be unsigned. If comparison occurs on all channels, the ADC_EMR.SIGNMODE field must be set to ALL_UNSIGNED or ALL_SIGNED and the thresholds must be set accordingly.

### 65.6.10 Differential and Single-ended Input Modes

#### 65.6.10.1 Input-output Transfer Functions

The ADC can be configured to operate in the following input voltage modes:

- Single-ended—ADC_COR.DIFFx = 0. This is the default mode after a reset.
- Differential—ADC_COR.DIFFx = 1 (see figure below). In Differential mode, the ADC requires differential input signals having a VDD/2 common mode voltage (refer to section “Electrical Characteristics”).
- The following equations give the unsigned ADC input-output transfer function in each mode(1). With signed conversions (see field ADC_EMR.SIGNMODE), subtract 2047 from the ADC_LCDR.DATA value given below.

Single-ended mode:

$$\text{ADC_LCDR.LDATA} = \frac{\text{ADx} - \text{GNDANA}}{\text{ADVREF} - \text{GNDANA}} \times 2^{12}$$

Differential mode:

$$\text{ADC_LCDR.LDATA} = \left(1 + \frac{\text{ADx} - \text{ADx+1}}{\text{ADVREF} - \text{GNDANA}}\right) \times 2^{11}$$

Note: 1. Equations assume ADC_EMR.OSR = 1

If the ADC_MR.ANACH bit is set, the ADC can manage both differential channels and single-ended channels. If the ADC_MR.ANACH bit is cleared, the parameters defined in ADC_COR are applied to all channels.

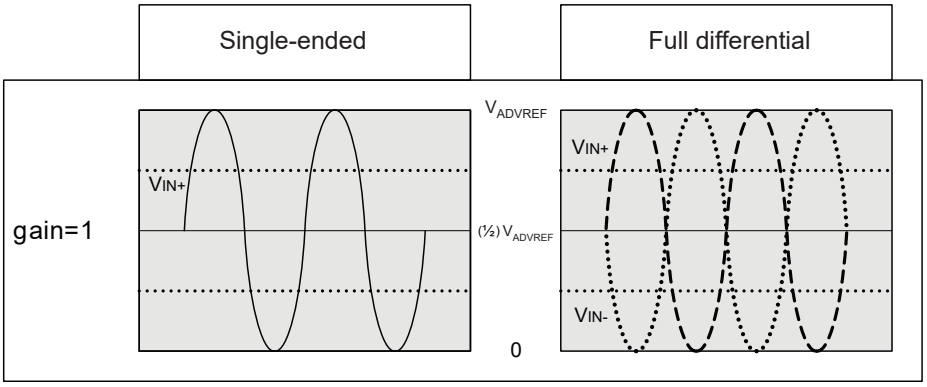
The following table gives s the internal positive and negative ADC inputs assignment with respect to the programmed mode (ADC_COR.DIFFx).

For example, if Differential mode is required on channel 0, input pins AD0 and AD1 are used. In this case, only channel 0 must be enabled by writing a 1 to ADC_CHER.CH0.

**Table 65-2. Input Pins and Channel Numbers**

| Input Pin | Channel Number | |
|---|---|---|
| | Single-ended Mode | Differential Mode |
| AD0 | CH0 | CH0 |
| AD1 | CH1 | |
| AD2 | CH2 | CH2 |
| AD3 | CH3 | |
| AD4 | CH4 | CH4 |
| AD5 | CH5 | |
| AD6 | CH6 | CH6 |
| AD7 | CH7 | |
| AD8 | CH8 | CH8 |
| AD9 | CH9 | |
| AD10 | CH10 | CH10 |
| AD11 | CH11 | |

**Figure 65-7. Analog Full Scale Ranges in Single-Ended/Differential Applications**




### 65.6.11 ADC Timings

The ADC startup time is programmed through the ADC_MR.STARTUP field. Refer to the “Electrical Characteristics” section.

The ADC Controller provides an inherent tracking time of six ADC clock cycles.

A minimal tracking time is necessary for the ADC to guarantee the best converted final value between two conversions. The tracking time can be adjusted to accommodate a range of source impedances. If more than six ADC clock cycles are required, the tracking time can be increased using the ADC_MR.TRACKTIM field.

**WARNING** No input buffer amplifier to isolate the source is included in the ADC. Refer to the “Electrical Characteristics” section.

### 65.6.12 Last Channel Specific Measurement Trigger

The last channel (higher index available) embeds a specific mode allowing a measurement trigger period which differs from other active channels. This allows efficient management of the conversions especially if the channel is driven by a device with a variation of a different frequency from other converted channels (for example, but not limited to, temperature sensor).

The last channel can be sampled in different ways through the ADC Controller. The different methods of sampling depend on the ADC_TRGR.TRGMOD configuration field and on the ADC_CHSR.CH11 bit.

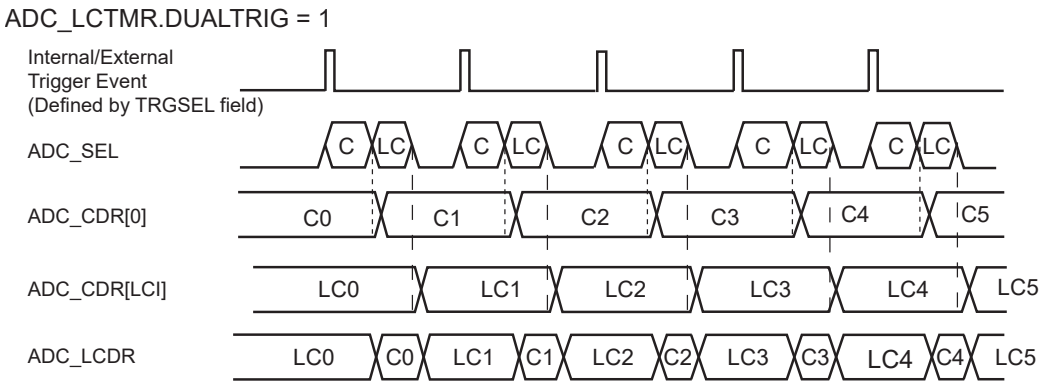
The last channel conversion can be triggered like the other channels by enabling the ADC_CHER.CH11 bit.

The manual start can only be performed if field TRGMOD = 0. When the ADC_CR.START bit is set, the last channel conversion is scheduled together with the other enabled channels (if any). The result of the conversion is placed in the ADC_CDR11 register, and the associated ADC_ISR.EOC11 flag is set.

If the last channel is enabled in the Channel Status register (ADC_CHSR), ADC_LCTMR.DUALTRIG is cleared and field TRGMOD = 1, 2, 3, 5, the last channel is periodically converted together with the other enabled channels and the result is placed in the ADC_LCDR and ADC_CDR11 registers. Thus the last channel conversion result is part of the DMA Controller buffer (see the following figure).

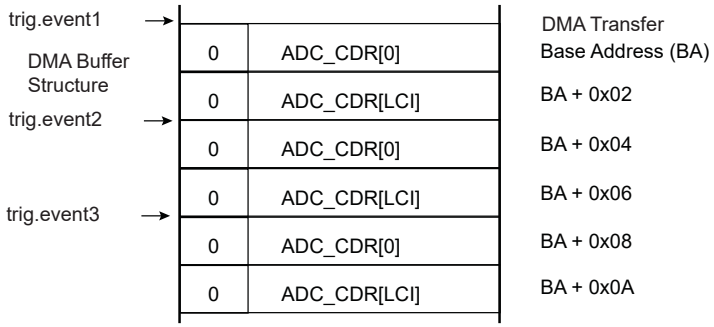
When the conversion result matches the conditions defined in ADC_LCTMR and ADC_LCCWR, the ADC_ISR.LCCHG flag is set.

**Figure 65-8. Same Trigger for All Channels (ADC_CHSR[LCI] = 1 and ADC_TRGR.TRGMOD = 1, 2, 3, 5)**




Notes: ADC_SEL: Command to the ADC analog cell
Cx: All ADC channel values except the last channel (highest index)
LCx: Last channel value
LCI: Last channel index

Assuming ADC_CHSR[0] = 1 and ADC_CHSR[LCI] = 1



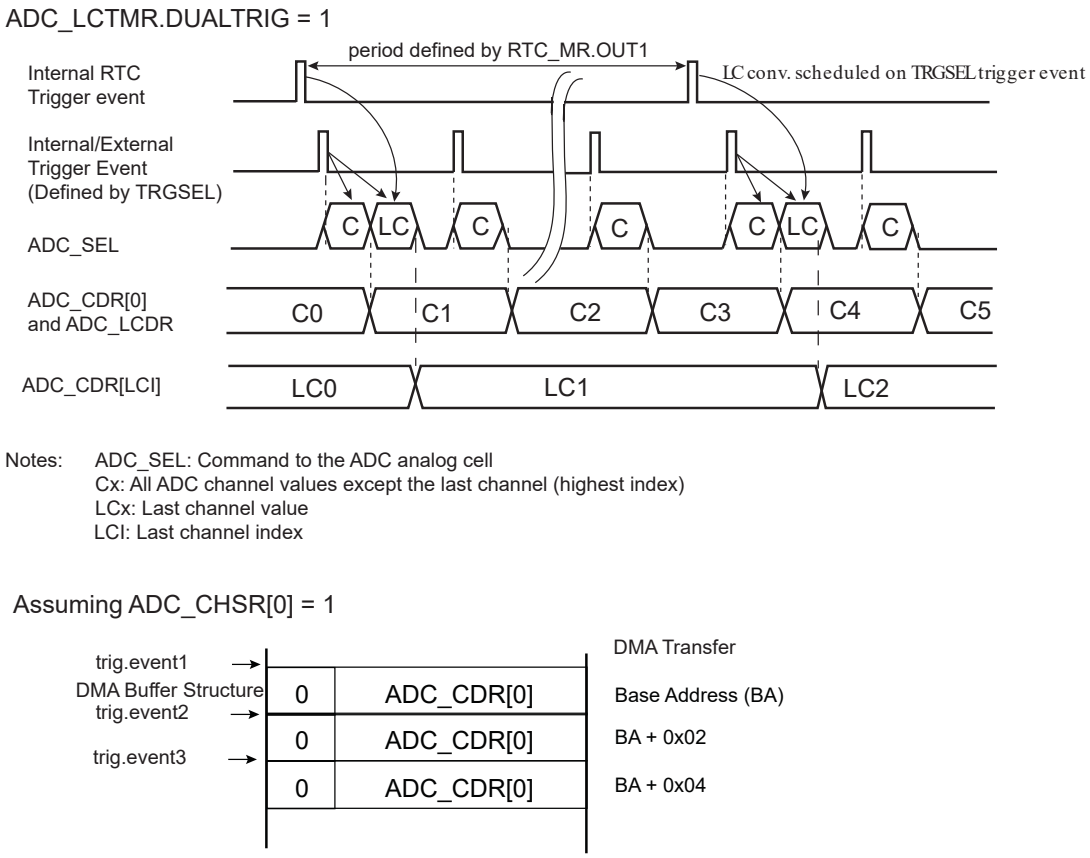

If the last channel is driven by a device with a slower variation compared to other channels (temperature sensor for example), the channel can be enabled/disabled at any time. However, this may not be optimal for downstream processing.

The ADC Controller allows a different way of triggering the measurement when DUALTRIG is set in the Last Channel Trigger Mode register (ADC_LCTMR) but CH11 is not set in ADC_CHSR.

Under these conditions, the last channel conversion is triggered with a period defined by the OUT1 field in the RTC_MR (Real-time Clock Mode register) while other channels are still active. OUT1 configures an internal trigger generated by the RTC, totally independent of the internal/external triggers. The RTC event will be processed on the next internal/external trigger event as shown in the following figure. The internal/external trigger for other channels is selected through the ADC_MR.TRGSEL field.

When DUALTRIG = 1, the result of each conversion of channel 11 is only uploaded in the ADC_CDR11 register and not in ADC_LCDR (see the following figure). Therefore, there is no change in the structure of the peripheral DMA controller buffer due to the conversion of the last channel: only the enabled channels are kept in the buffer. The end of conversion of the last channel is reported by the ADC_ISR.EOC11 flag.

**Figure 65-9. Independent Trigger Measurement for Last Channel (ADC_CHSR[LCI] = 0 and ADC_TRGR.TRGMOD = 1, 2, 3, 5)**

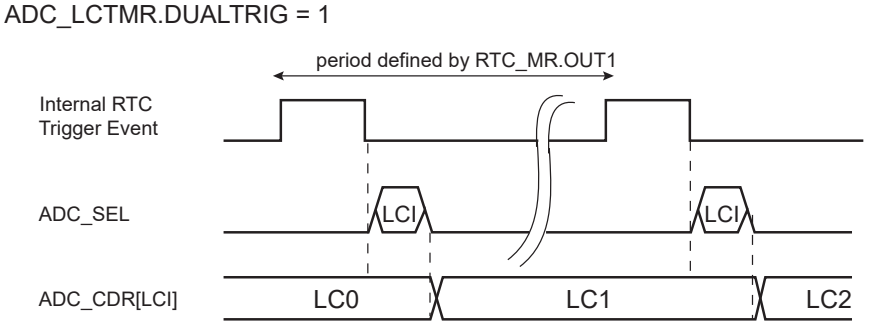



Notes: ADC_SEL: Command to the ADC analog cell
Cx: All ADC channel values except the last channel (highest index)
LCx: Last channel value
LCI: Last channel index

If DUALTRIG = 1 and field ADC_TRGR.TRGMOD = 0 and none of the channels are enabled in ADC_CHSR (ADC_CHSR = 0), then only channel 11 is converted at a rate defined by the trigger event signal that can be configured in RTC_MR.OUT1 (see the following figure).

This mode of operation, when combined with the Sleep mode operation of the ADC Controller, provides a low-power mode for last channel measure. This assumes there is no other ADC conversion to schedule at a high sampling rate or no other channel to convert.

**Figure 65-10. Only Last Channel Measurement Triggered at Low Speed (ADC_CHSR[LCI] = 0 and ADC_TRGR.TRGMOD = 0)**




Notes: ADC_SEL: Command to the ADC analog cell
LCx: Last channel value
LCI: Last channel index

### 65.6.13 Enhanced Resolution Mode and Digital Averaging Function

#### 65.6.13.1 Enhanced Resolution Mode

The Enhanced Resolution mode is enabled if the OSR field is configured to 1 or 2 in ADC_EMR. The enhancement is based on a digital averaging function.

There is no averaging on the last index channel if the measure is triggered by an RTC event.

In this mode, the ADC Controller will trade off conversion speed against accuracy by averaging multiple samples, thus providing a digital low-pass filter function.

The selected oversampling ratio applies to all enabled channels when triggered by an RTC event.

$$\text{ADC_LCDR.LDATA} = \frac{1}{M} \times \sum_{k=0}^{k=N-1} \text{ADC}(k)$$

where N and M are given in the table below.

**Table 65-3. Digital Averaging Function Configuration versus OSR Values**

| ADC_EMR.OSR Value | ADC_LCDR.LDATA Length | N Value | M Value | Full Scale Value | Maximum Value |
|---|---|---|---|---|---|
| 0 | 12 bits | 1 | 1 | 4095 | 4095 |
| 1 | 13 bits | 4 | 2 | 8191 | 8190 |
| 2 | 14 bits | 16 | 4 | 16383 | 16381 |

The average result is valid in ADC_CDRx (x corresponds to the index of the channel) only if the ADC_ISR.EOCn flag is set and if the ADC_OVER.OVREn flag is cleared. The average result for all channels is valid in ADC_LCDR only if the ADC_ISR.DRDY bit is set and the ADC_ISR.GOVRE bit is cleared.

Note that ADC_CDRs are not buffered. Therefore, when an averaging sequence is ongoing, the value in these registers changes after each averaging sample. However, overrun flags in ADC_OVER rise as soon as the first sample of an averaging sequence is received. Thus the previous averaged value is not read, even if the new averaged value is not ready.

Consequently, when an overrun flag rises in ADC_OVER, it means that the previous unread data is lost but it does not mean that this data has been overwritten by the new averaged value as the averaging sequence concerning this channel can still be ongoing.

When an oversampling is performed, the maximum value that can be read on ADC_CDRx or ADC_LCDR is not the full-scale value, even if the maximum voltage is supplied on the analog input. See table above.

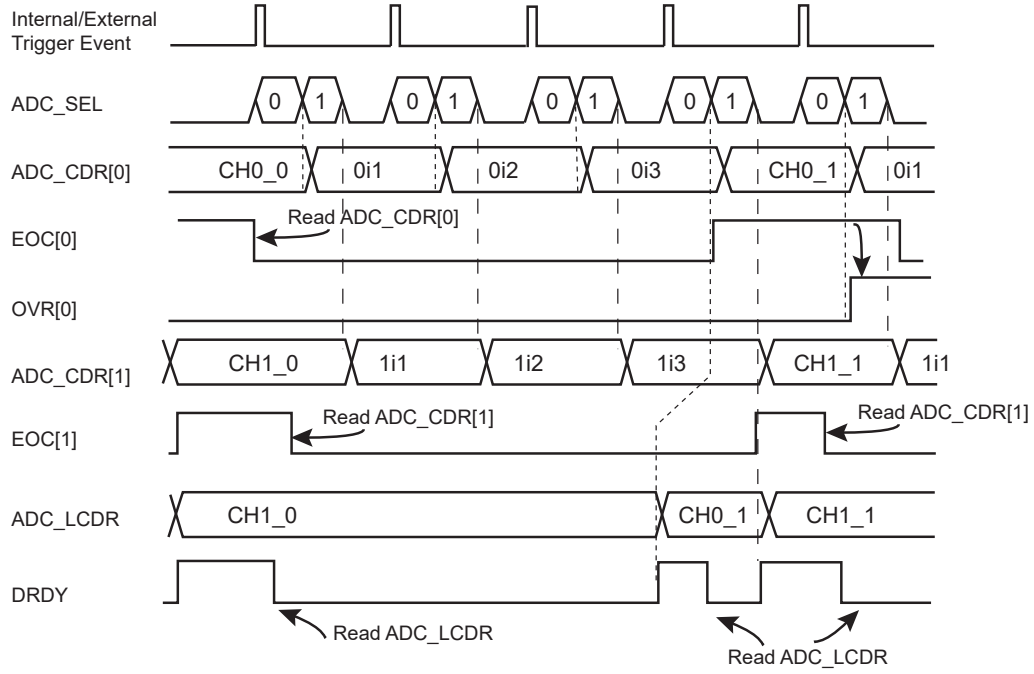
#### 65.6.13.2 Averaging Function versus Trigger Events

The samples can be defined in different ways for the averaging function depending on the configuration of the ADC_EMR.ASTE bit and the ADC_MR.USEQ bit.

When USEQ = 0, there are two possible ways to generate the averaging through the trigger event. If ADC_EMR.ASTE = 0, every trigger event generates one sample for each enabled channel, as described in the following figure. Therefore, four trigger events are required to obtain the result of averaging if OSR = 1.

**Figure 65-11. Digital Averaging Function Waveforms Over Multiple Trigger Events**

ADC_EMR.OSR = 1, ASTE = 0, ADC_CHSR[1:0] = 0x3 and ADC_MR.USEQ = 0

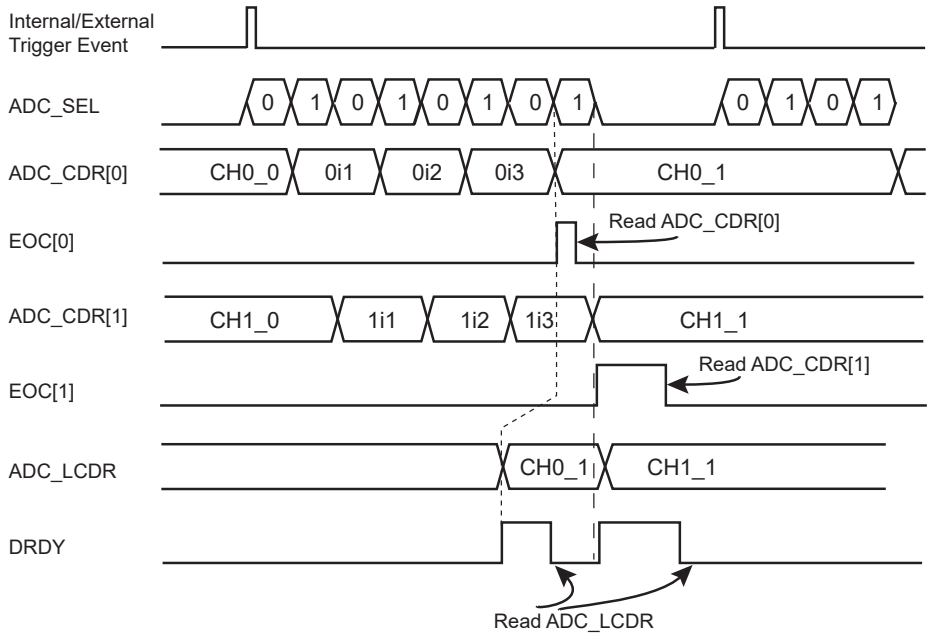



Note: ADC_SEL: Command to the ADC analog cell
0i1, 0i2, 0i3, 1i1, 1i2, 1i3 are intermediate results and CH0_0, CH0_1, CH1_0 and CH1_1 are final results of average function.

If ADC_EMR.ASTE = 1 and ADC_MR.USEQ = 0, the sequence to be converted, defined in ADC_CHSR, is automatically repeated n times (where n corresponds to the oversampling ratio defined in the ADC_EMR.OSR field). As a result, only one trigger is required to obtain the result of the averaging function as described in the following figure.

**Figure 65-12. Digital Averaging Function Waveforms on a Single Trigger Event**

ADC_EMR.OSR = 1, ASTE = 1, ADC_CHSR[1:0] = 0x3 and ADC_MR.USEQ = 0




Note: ADC_SEL: Command to the ADC analog cell
0i1, 0i2, 0i3, 1i1, 1i2, 1i3 are intermediate results and CH0_0, CH0_1, CH1_0 and CH1_1 are final results of average function.

When USEQ = 1, the user can define the channel sequence to be converted by configuring ADC_SEQRx and ADC_CHER so that channels are not interleaved during the averaging period. Under these conditions, a sample is defined for each end of conversion as described in the figure below.

When USEQ = 1 and ASTE = 1, OSR can be only configured to 1. Up to three channels can be converted in this mode. The averaging result will be placed in the corresponding ADC_CDRx and in ADC_LCDR for each trigger event. The ADC real sample rate remains the maximum ADC sample rate divided by 4.

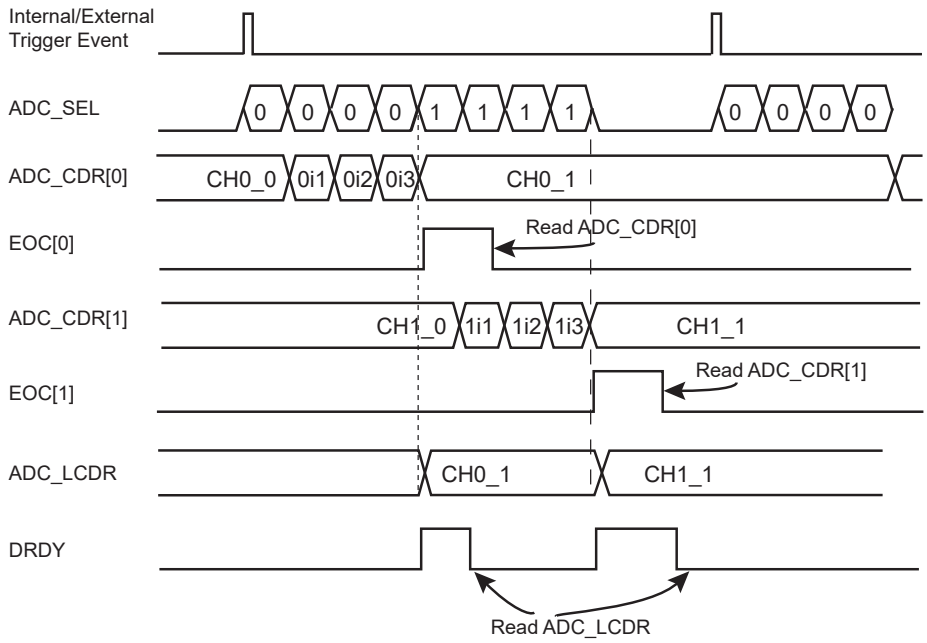
It is important that the user sequence follows a specific pattern. The user sequence must be programmed in such a way that it generates a stream of conversion, where a same channel is successively converted.

**Table 65-4. Example Sequence Configurations (USEQ = 1, ASTE = 1, OSR = 1)**

| Register | Number of Channels Non-interleaved Averaging - Register Value | | |
|---|---|---|---|
| | 1 (e.g., CH0) | 2 (e.g., CH0, CH1) | 3 (e.g., CH0, CH1, CH2) |
| ADC_CHSR | 0x0000_000F | 0x0000_00FF | 0x0000_0FFF |
| ADC_SEQR1 | 0x0000_0000 | 0x1111_0000 | 0x1111_0000 |
| ADC_SEQR2 | 0x0000_0000 | 0x0000_0000 | 0x0000_2222 |

**Figure 65-13. Digital Averaging Function Waveforms on a Single Trigger Event, Non-interleaved**

ADC_EMR.OSR = 1, ASTE = 1, ADC_CHSR[7:0] = 0xFF and ADC_MR.USEQ = 1
ADC_SEQR1 = 0x1111_0000




Note: ADC_SEL: Command to the ADC analog cell
0i1, 0i2, 0i3, 1i1, 1i2, 1i3 are intermediate results and CH0_0, CH0_1, CH1_0 and CH1_1 are final results of average function.

### 65.6.14 Automatic Error Correction

The ADC features automatic error correction of conversion results. Offset and gain error corrections are available. The correction can be enabled for each channel and correction values (offset and gain) are the same for all channels.

To enable error correction, the corresponding ECORRx bit must be set in the Channel Error Correction register (ADC_CECR). The offset and gain values used to compensate the results are the same for all correction-enabled channels and programmed in the Correction Values register (ADC_CVR).

The error correction for channels used with the touchscreen is available in the ADC Touchscreen Correction Values register (ADC_TSCVR).

The ADC_EMR.ADCMODE field is used to configure a running mode of the ADC Normal mode, Offset Error mode, or Gain Error mode (see 65.7.16 ADC_EMR). ADCMODE uses 3 internal references to be measured and to extract the offset and gain error from 3 point-measurement codes. If some references already exist on the final application connected to some input channel ADx, they can be used as a replacement of the ADCMODE to generate the 2 or 3 points of calibration and used to extract the GAINCORR and OFFSETCORR.

After a reset, the running mode of the ADC is Normal mode. Offset Error mode and Gain Error mode are used to determine values of offset compensation and gain compensation, respectively, to apply to conversion results. The table below provides formulas to obtain the compensation values, with:

- OFFSETCORR—the Offset Correction value. OFFSETCORR is a signed value.
- GAINCORR—the Gain Correction value
- GCi—the intermediate Gain Compensation value

- Gs—the value 13
- ConvValue—the value converted by the ADC (as returned in ADC_LCDR or ADC_CDR)
- Resolution—the resolution used to process the conversion (either RESOLUTION, RESOLUTION+1 or RESOLUTION+2).

**Table 65-5. ADC Running Modes**

| ADC_EMR.ADCMODE | Mode | Description |
|---|---|---|
| 0 | Normal | Normal mode of operation to perform conversions |
| 1 | Offset Error | For unsigned conversions: OFFSETCORR = ConvValue – $2^{(\text{Resolution}} - 1)$ |
| | | For signed conversions: OFFSETCORR = ConvValue |
| 2 | Gain Error | GCi = ConvValue |
| 3 | | $\text{GAINCORR} = \frac{3584}{\text{GCi} - \text{ConvValue}} \times 2^{(\text{Gs})}$ |

The final conversion result after error correction is obtained using the following formula:

$$\text{Corrected Data} = (\text{Converted Data} + \text{OFFSETCORR}) \times \frac{\text{GAINCORR}}{2^{(\text{Gs})}}$$

### 65.6.15 Touchscreen

#### 65.6.15.1 Touchscreen Mode

The ADC_TSMR.TSMODE parameter is used to enable/disable the touchscreen functionality, to select the type of screen (4-wire or 5-wire) and, in the case of a 4-wire screen, to activate (or not) the pressure measurement.

In 4-wire mode, channels 0, 1, 2 and 3 must not be used for classic ADC conversions. Likewise, in 5-wire mode, channels 0, 1, 2, 3, and 4 must not be used for classic ADC conversions.

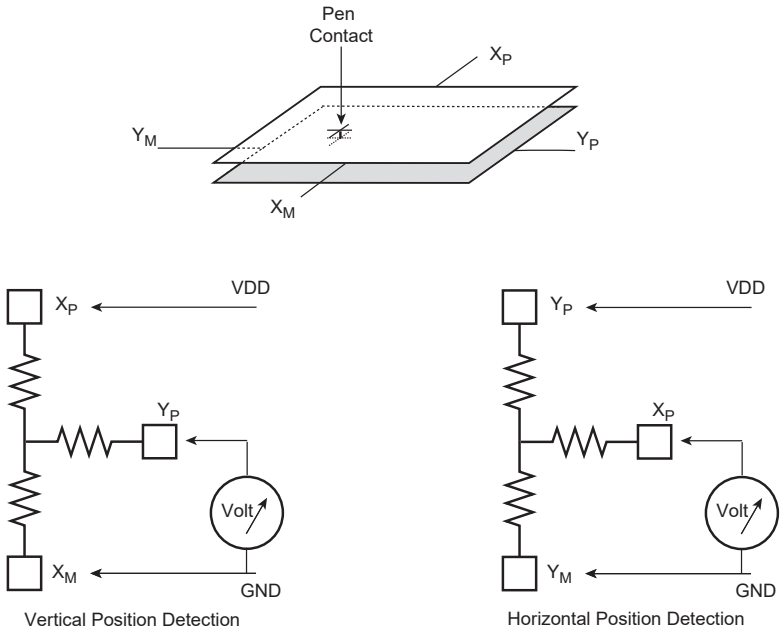
#### 65.6.15.2 4-wire Resistive Touchscreen Principles

A resistive touchscreen is based on two resistive films, each one being fitted with a pair of electrodes, placed at the top and bottom on one film, and on the right and left on the other. In between, there is a layer acting as an insulator, but also enables contact when you press the screen. This is illustrated in the following figure.

The ADC Controller can perform the following tasks without external components:

- position measurement
- pressure measurement
- pen detection

**Figure 65-14. Touchscreen Position Measurement**




#### 65.6.15.3 4-wire Position Measurement Method

As shown in the above figure, to detect the position of a contact, a supply is first applied from top to bottom. Due to the linear resistance of the film, there is a voltage gradient from top to bottom. When a contact is performed on the screen, the voltage propagates at the point the two surfaces come into contact. If the input impedance on the right and left electrodes is high enough, the film intrinsic resistor does not affect this voltage.

For the horizontal direction, the same method is used, but by applying supply from left to right. The range depends on the supply voltage and on the loss in the switches that connect to the top and bottom electrodes.

In an ideal world (linear, with no loss through switches), the horizontal position is equal to:

$VY_M$ / VDD or $VY_P$ / VDD.

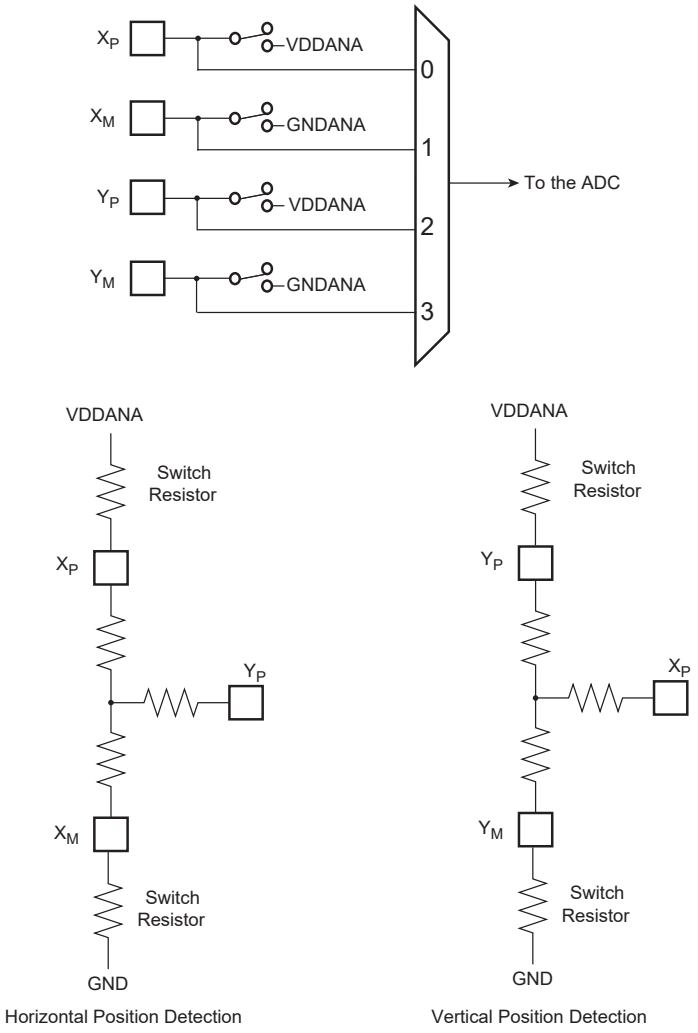
The implementation with on-chip power switches is shown in the figure below. The voltage measurement at the output of the switch compensates for the switches loss.

It is possible to correct for switch loss by performing the operation:

[$VY_P$ - $VX_M$] / [$VX_P$ - $VX_M$].

This requires additional measurements, as shown in the figure below.

**Figure 65-15. Touchscreen Switches Implementation**



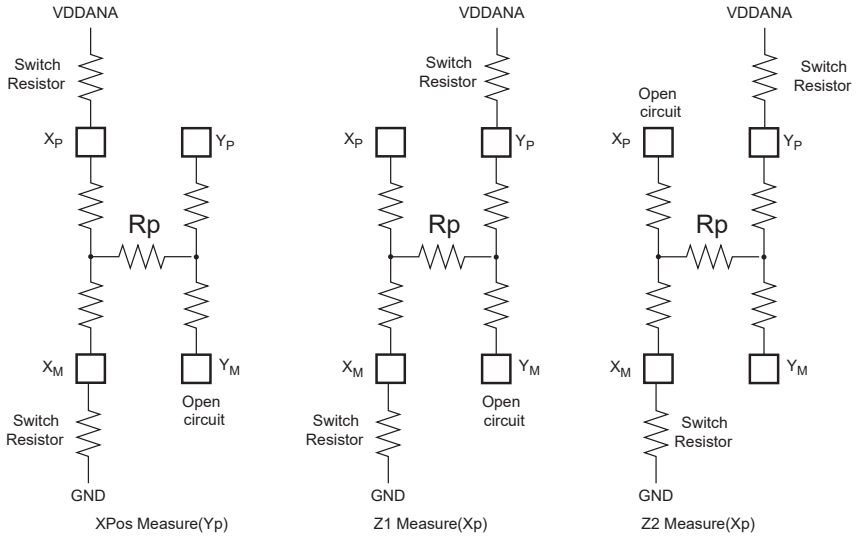

#### 65.6.15.4 4-wire Pressure Measurement Method

The method to measure the pressure (Rp) applied to the touchscreen is based on the known resistance of the X-Panel resistance (Rxp).

Three conversions (Xpos,Z1,Z2) are necessary to determine the value of Rp (Zaxis resistance).

$Rp = Rxp \times (Xpos/1024) \times [(Z2/Z1)-1]$

**Figure 65-16. Pressure Measurement**




### 65.6.15.5 5-wire Resistive Touchscreen Principles

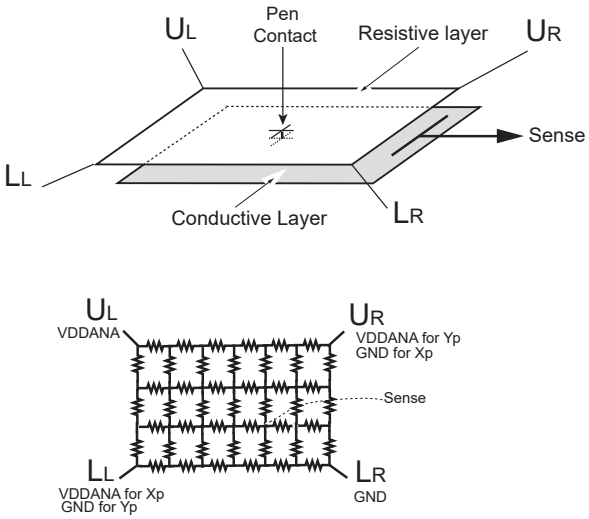
To make a 5-wire touchscreen, a resistive layer with a contact point at each corner and a conductive layer are used.

The 5-wire touchscreen differs from the 4-wire type mainly in that the voltage gradient is applied only to one layer, the resistive layer, while the other layer is the sense layer for both measurements.

The measurement of the X position is obtained by biasing the upper left corner and lower left corner to VDDANA and the upper right corner and lower right to ground.

To measure along the Y axis, bias the upper left corner and upper right corner to VDDANA and bias the lower left corner and lower right corner to ground.

**Figure 65-17. 5-Wire Principle**




### 65.6.15.6 5-wire Position Measurement Method

In an application only monitoring clicks, 100 points per second is typically needed. For handwriting or motion detection, the number of measurements to consider is approximately 200 points per second. This must take into account that multiple measurements are included (over sampling, filtering) to compute the correct point.

The 5-wire touchscreen panel works by applying a voltage at the corners of the resistive layer and measuring the vertical or horizontal resistive network with the sense input. The ADC converts the voltage measured at the point the panel is touched.

A measurement of the Y position of the pointing device is made by:

- Connecting Upper left (UL) and upper right (UR) corners to VDDANA
- Connecting Lower left (LL) and lower right (LR) corners to ground.

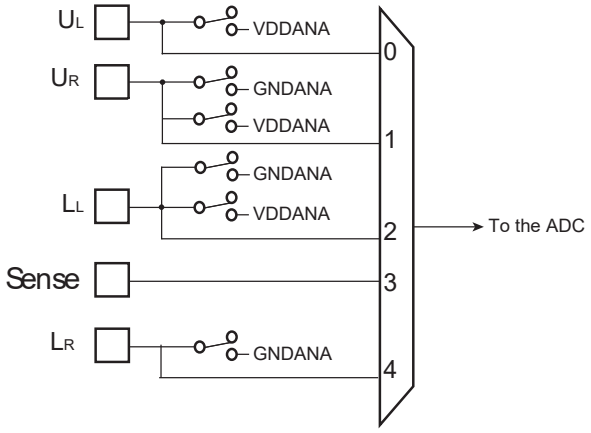
The voltage measured is determined by the voltage divider developed at the point of touch (Y position) and the SENSE input is converted by ADC.

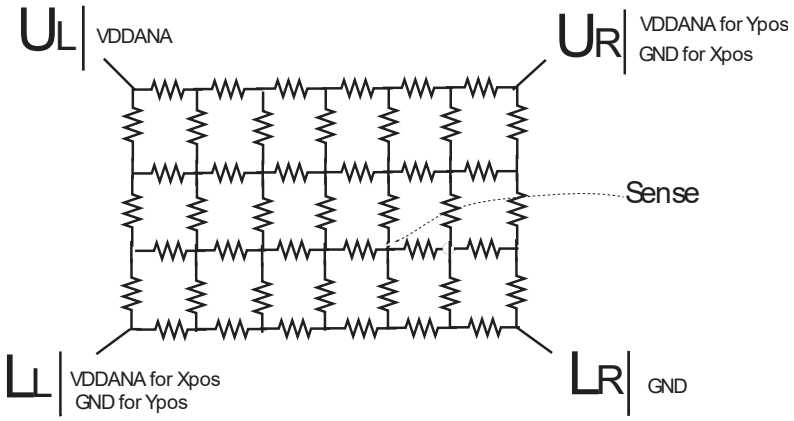
A measurement of the X position of the pointing device is made by:

- Connecting the upper left (UL) and lower left (LL) corners to ground
- Connecting the upper right and lower right corners to VDDANA.

The voltage measured is determined by the voltage divider developed at the point of touch (X position) and the SENSE input is converted by ADC.

**Figure 65-18. Touchscreen Switches Implementation**



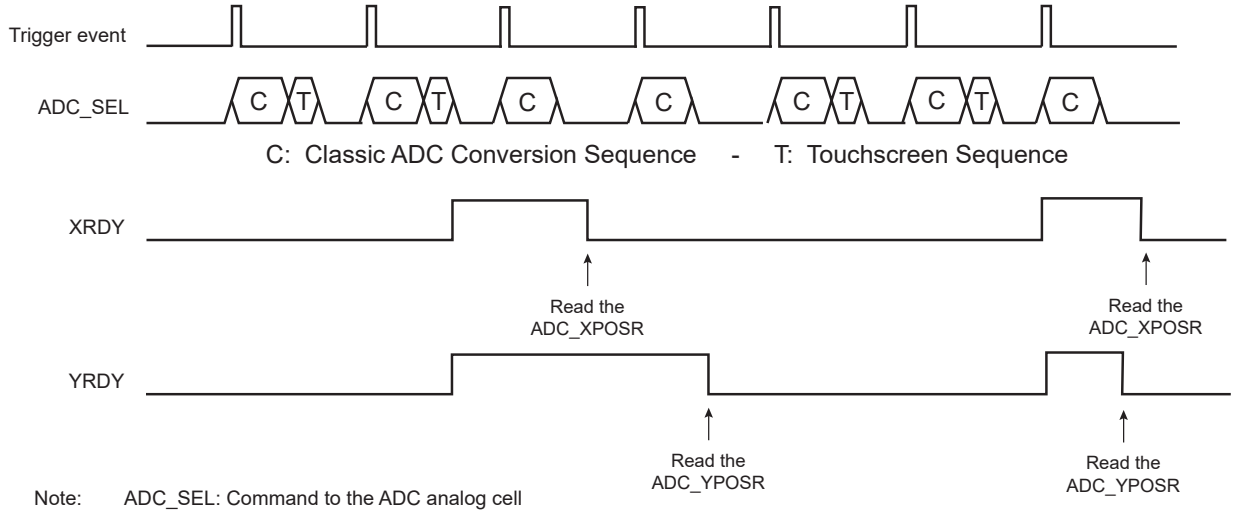

### 65.6.15.7 Sequence and Noise Filtering

The ADC Controller can manage ADC conversions and touchscreen measurement. On each trigger event the sequence of ADC conversions is performed as described in 65.6.8 Sleep Mode and Conversion Sequencer. The touchscreen measure frequency can be specified in number of trigger events

by writing the ADC_TSMR.TSFREQ parameter. An internal counter counts triggers up to TSFREQ, and every time it rolls out, a touchscreen sequence is appended to the classic ADC conversion sequence (see figure below).

Additionally the user can average multiple touchscreen measures by writing the ADC_TSMR.TSAV parameter. This can be 1, 2, 4 or 8 measures performed on consecutive triggers as illustrated in the figure below. Consequently, the ADC_TSMR.TSFREQ parameter must be greater than or equal to the ADC_TSMR.TSAV parameter.

**Figure 65-19. Insertion of Touchscreen Sequences (TSFREQ = 2; TSAV = 1)**




### 65.6.15.8 Measured Values, Registers and Flags

As soon as the controller finishes the Touchscreen sequence, XRDY, YRDY and PRDY are set and can generate an interrupt. These flags can be read in the Interrupt Status register (ADC_ISR). They are reset independently by reading in the ADC Touchscreen X Position register (ADC_XPOSR), the ADC Touchscreen Y Position register (ADC_YPOSR) and the ADC Touchscreen Pressure register (ADC_PRESSR).

ADC_XPOSR presents XPOS (VX - VXmin) on its LSB and XSCALE (VXMAX - VXmin) aligned on the 16th bit.

ADC_YPOSR presents YPOS (VY - VYmin) on its LSB and YSCALE (VYMAX - VYmin) aligned on the 16th bit.

To improve the quality of the measure, the user must calculate XPOS/XSCALE and YPOS/YSCALE.

VXMAX, VXmin, VYMAX, and VYmin are measured at the first startup of the controller. These values can change during use, so it can be necessary to refresh them. Refresh can be done by writing ‘1’ in the ADC_CR.TSCALIB field.

ADC_PRESSR presents Z1 on its LSB and Z2 aligned on the 16th bit. See 65.6.15.4 4-wire Pressure Measurement Method.

### 65.6.15.9 Pen Detect Method

When there is no contact, it is not necessary to perform a conversion. However, it is important to detect a contact by keeping the power consumption as low as possible.

The implementation polarizes one panel by closing the switch on ($X_P$/$U_L$) and ties the horizontal panel by an embedded resistor connected to $Y_M$ / Sense. This resistor is enabled by a fifth switch. Since there is

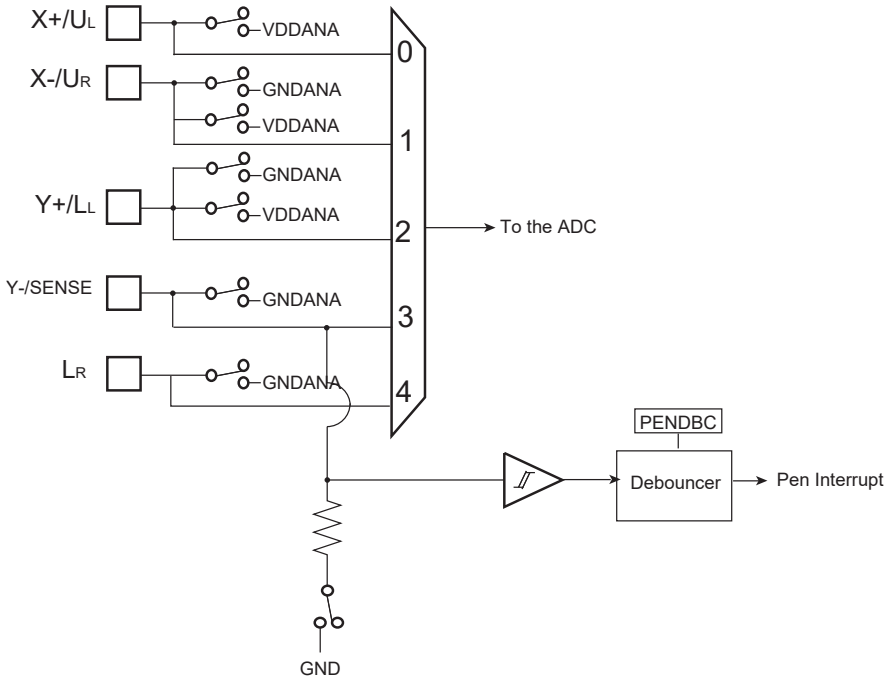
no contact, no current is flowing and there is no related power consumption. As soon as a contact occurs, a current is flowing in the Touchscreen and a Schmitt trigger detects the voltage in the resistor.

The Touchscreen Interrupt configuration is entered by programming the ADC_TSMR.PENDET bit. If this bit is written at 1, the controller samples the pen contact state when it is not converting and waiting for a trigger.

To complete the circuit, a programmable debouncer is placed at the output of the Schmitt trigger. This debouncer is programmable up to $2^{15}$ ADC clock periods. The debouncer length can be selected by programming the ADC_TSMR.PENDBC field.

Due to the analog switch's structure, the debouncer circuitry is only active when no conversion (touchscreen or classic ADC channels) is in progress. Thus, if the time between the end of a conversion sequence and the arrival of the next trigger event is lower than the debouncing time configured on ADC_TSMR.PENDBC, the debouncer will not detect any contact.

**Figure 65-20. Touchscreen Pen Detect**




The touchscreen pen detect can be used to generate an ADC interrupt to wake up the system. The pen detect generates two types of status, reported in ADC_ISR:

- the ADC_ISR.PEN bit is set as soon as a contact exceeds the debouncing time as defined by the ADC_TSMR.PENDBC field and remains set until ADC_ISR is read.
- the ADC_ISR.NOPEN bit is set as soon as no current flows for a time over the debouncing time as defined by PENDBC and remains set until ADC_ISR is read.

Both bits are automatically cleared as soon as ADC_ISR is read, and can generate an interrupt by writing ADC_IER.

Moreover, the rising of either one of them clears the other, they cannot be set at the same time.

The ADC_ISR.PENS bit shows the current status of the pen contact.

### 65.6.16 Asynchronous and Partial Wakeup (SleepWalking)

This operating mode is a means of data preprocessing that qualifies an incoming event, thus allowing the ADC to decide whether or not to wake up the system. Asynchronous and partial wakeup is mainly used when the system is in Wait mode (refer to the Power Management Controller (PMC) section for further details). It can also be enabled when the system is fully running.

Once the Asynchronous and partial wakeup mode is enabled, no access must be performed in the ADC before a wakeup is performed by the ADC.

When the Asynchronous and partial wakeup mode is enabled for the ADC (refer to the Power Management Controller (PMC) section), the PMC decodes a clock request from the ADC. The clock request is generated as soon as a trigger event occurs. Only a trigger from RTC or ADTRG pin can be used in partial wakeup mode. The selection between RTC or ADTRG pin is performed through the ADC_MR.TRGSEL field.

If the system is in Wait mode (processor and peripheral clocks switched off), the PMC restarts the fast RC oscillator and provides the clock only to the ADC.

To perform a conversion at regular intervals with RTC trigger, the RTC must be configured with the following settings: RTC_MR.OUT0=7 and RTC_MR.THIGH=7. The period of the trigger can be defined in RTC_MR.TPERIOD.

To trigger a conversion using the ADTRG pin, the minimum high level duration of the ADTRG signal must be greater than 2 clock periods of the fast RC oscillator. The maximum duration of the high level must be limited to the amount of startup and conversion time.

As soon as the clock is provided by the PMC, the ADC processes the conversions and compares the converted values with the ADC_CWR.LOWTHRES and ADC_CWR.HIGHTHRES field values.

The ADC instructs the PMC to disable the clock if the converted value does not meet the conditions defined by the ADC_CWR.LOWTHRES and ADC_CWR.HIGHTHRES field values.

If the converted value meets the conditions, the ADC instructs the PMC to exit the full system from Wait mode.

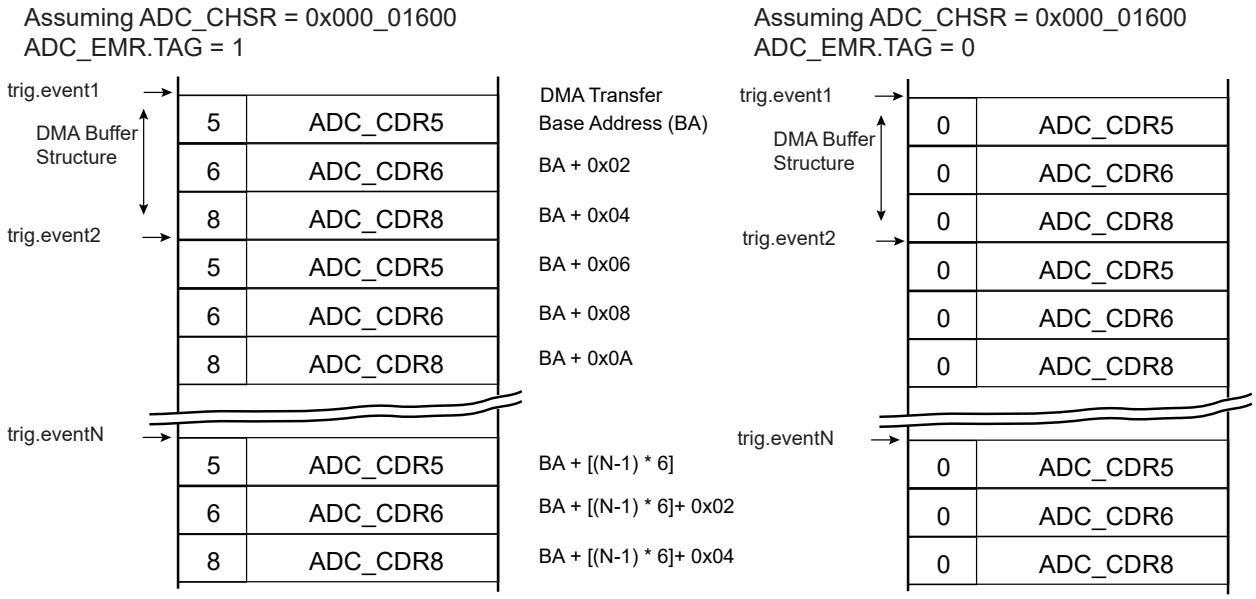
If the processor and peripherals are running, the ADC can be configured in Asynchronous and partial wakeup mode by enabling PMC_SLPWK_ER (refer to the Power Management Controller (PMC) section). When a trigger event occurs, the ADC requests the clock from the PMC and the comparison is performed. If there is a comparison match, the ADC continues to request the clock. If there is no match, the clock is switched off for the ADC only, until a new trigger event is detected.

It is recommended to write a ‘1’ to the ADC_MR.SLEEP bit to reduce the power consumption of the ADC analog part when the system is waiting for a trigger event.

### 65.6.17 Buffer Structure

The DMA read channel is triggered each time a new data is stored in ADC_LCDR. The same data structure is repeatedly stored in ADC_LCDR each time a trigger event occurs. Depending on user mode of operation (ADC_MR, ADC_CHSR, ADC_SEQR1, ADC_SEQR2, ADC_TSMR) the structure differs. Each data read to DMA buffer, carried on a half-word (16-bit), consists of last converted data right-aligned and when the ADC_EMR.TAG is set, the four most significant bits are carrying the channel number thus allowing an easier postprocessing in the DMA buffer or better checking the DMA buffer integrity.

**Figure 65-21. Buffer Structure**




As soon as touchscreen conversions are required, the pen detection function can help the postprocessing of the buffer. See 65.6.17.4 Pen Detection Status.

### 65.6.17.1 Classic ADC Channels Only (Touchscreen Disabled)

When no touchscreen conversion is required (i.e., ADC_TSMR.TSMODE = 0), the data structure within the buffer is defined by ADC_MR, ADC_CHSR, ADC_SEQRx. See figure Buffer Structure.

If the user sequence is not used (i.e., ADC_MR.USEQ is cleared) then only the value of ADC_CHSR defines the data structure. For each trigger event, enabled channels will be consecutively stored in ADC_LCDR and automatically read to the buffer.

When the user sequence is configured (i.e., ADC_MR.USEQ is set) not only does ADC_CHSR modify the data structure of the buffer, but ADC_SEQRx registers may modify the data structure of the buffer as well.

### 65.6.17.2 Touchscreen Channels Only

When only touchscreen conversions are required (i.e., TSMODE ≠ 0 in ADC_TSMR and ADC_CHSR equals 0), the structure of data within the buffer is defined by ADC_TSMR.

When TSMODE = 1 or 3, each trigger event adds two half-words in the buffer (assuming TSAV = 0), first half-word being ADC_XPOSR.XPOS, then ADC_YPOSR.YPOS. If TSAV/TSFREQ ≠ 0, the data structure remains unchanged. Not all trigger events add data to the buffer.

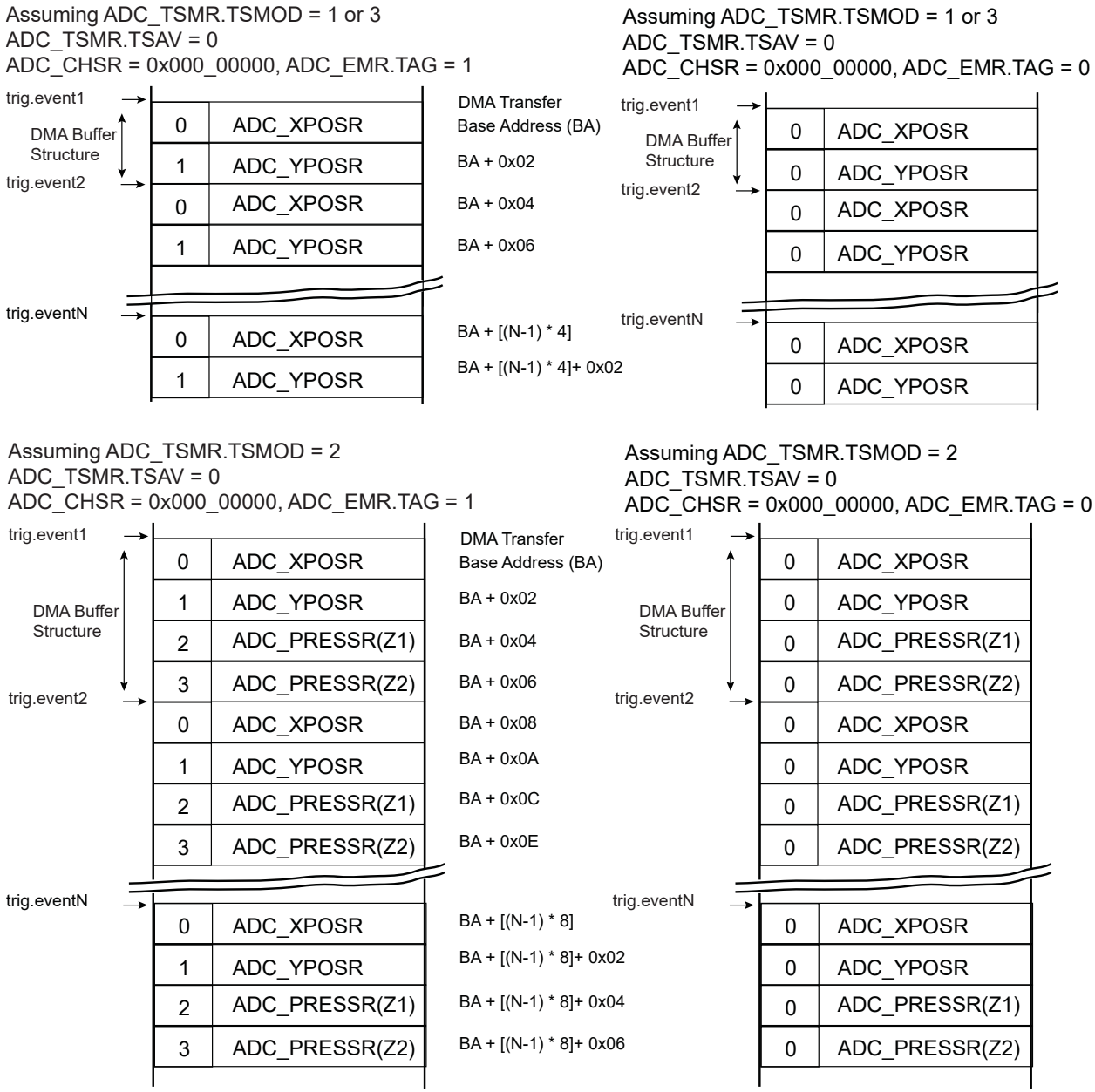
When TSMODE = 2, each trigger event adds four half-words to the buffer (assuming TSAV = 0), first half-word being ADC_XPOSR.XPOS, followed by ADC_YPOSR.YPOS and finally ADC_PRESSR.Z1, followed by ADC_PRESSR.Z2.

When ADC_EMR.TAG is set, the CHNB field (four most significant bits of ADC_LCDR) is cleared when ADC_XPOSR.XPOS is transmitted and set when ADC_YPOSR.YPOS is transmitted, allowing an easier post-processing of the buffer or a better checking of the buffer integrity. In case 4-wire with Pressure mode is selected, the Z1 value is transmitted to the buffer along with tag set to 2 and Z2 is tagged with value 3.

XSCALE and YSCALE (calibration values) are not transmitted to the buffer because they are supposed to be constant and moreover only measured at the very first startup of the controller or upon user request.

There is no change in buffer structure whatever the value of the PENDET.ADC_TSMR bit configuration, but it is recommended to use the pen detection function for buffer postprocessing (see 65.6.17.4 Pen Detection Status).

**Figure 65-22. Buffer Structure When Only Touchscreen Channels are Enabled**




### 65.6.17.3 Interleaved Channels

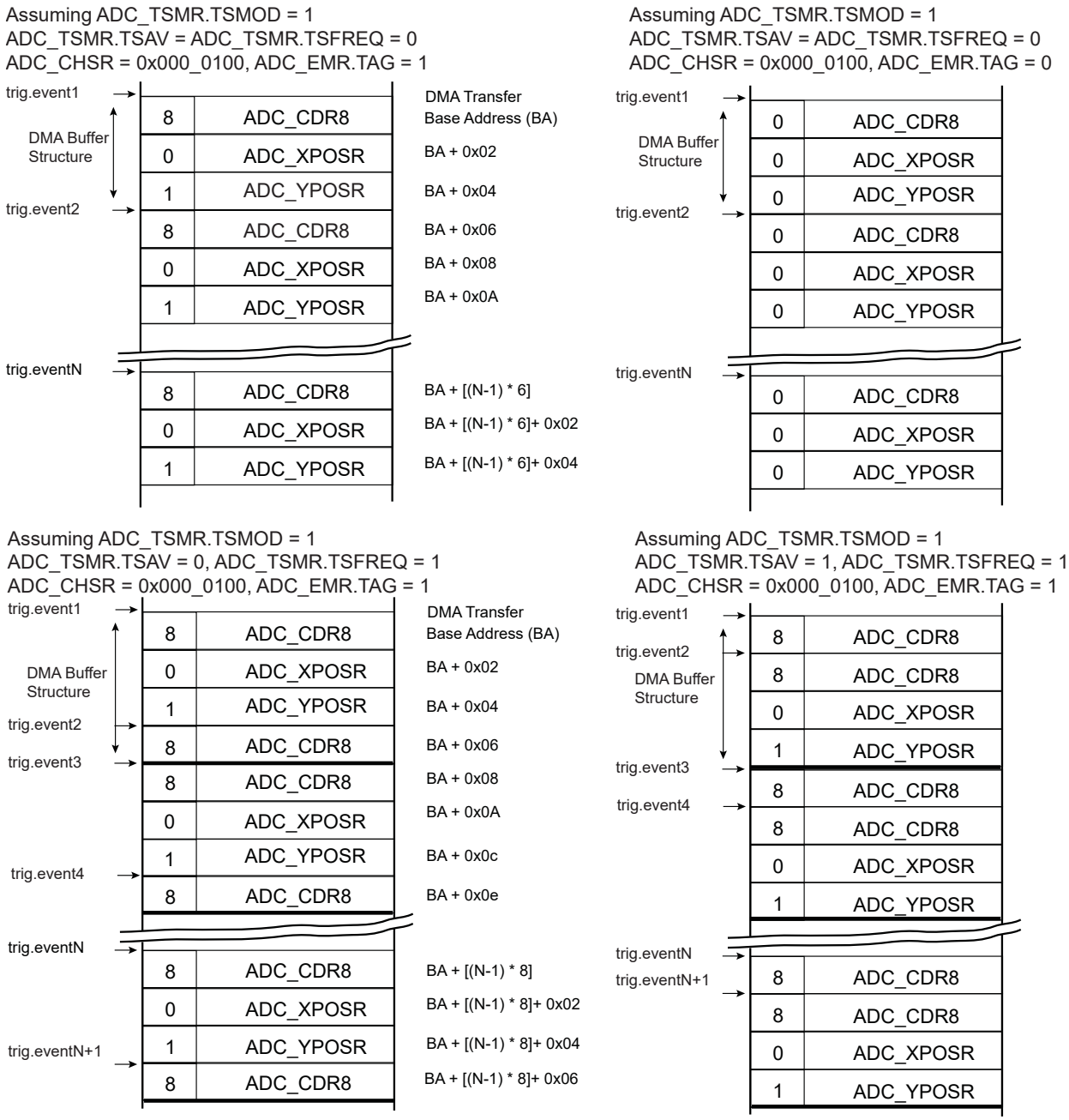
When both classic ADC channels (CH4/CH5 up to CH12 are set in ADC_CHSR) and touchscreen conversions are required (TSMODE ≠ 0 in ADC_TSMR), the structure of the buffer differs according to the ADC_TSMR.TSAV and ADC_TSMR.TSFREQ values.

If TSFREQ ≠ 0, not all events generate touchscreen conversions, therefore the buffer structure is based on $2^{TSFREQ}$ trigger events. Given a TSFREQ value, the location of touchscreen conversion results depends on TSAV value.

When TSFREQ = 0, TSAV must equal 0.

There is no change in buffer structure whatever the value of the ADC_TSMR.PENDET bit configuration, but it is recommended to use the pen detection function for buffer post-processing (see 65.6.17.4 Pen Detection Status).

**Figure 65-23. Buffer Structure When Classic ADC and Touchscreen Channels are Interleaved**



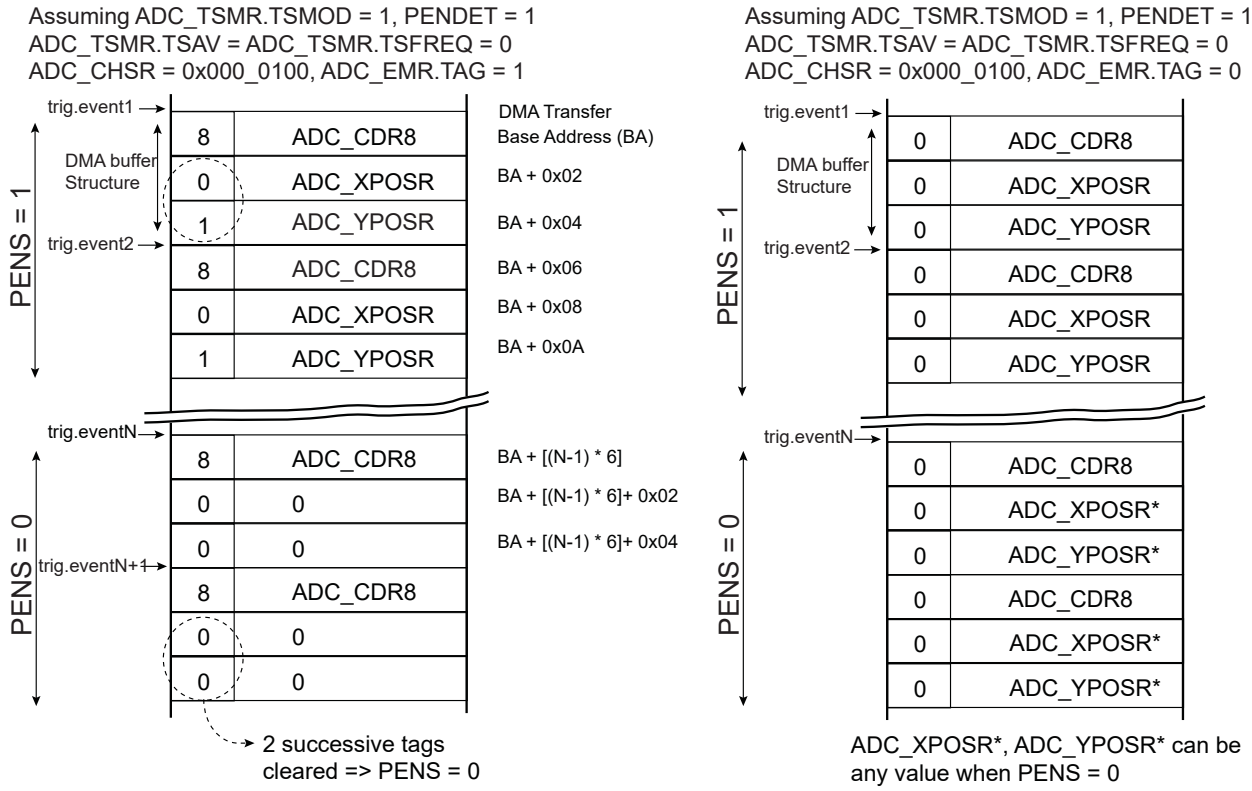

### 65.6.17.4 Pen Detection Status

If the pen detection measure is enabled (ADC_TSMR.PENDET is set), the XPOS, YPOS, Z1, Z2 values transmitted to the buffer through ADC_LCDR are cleared (including the CHNB field), if the ADC_ISR.PENS flag is 0. When the ADC_ISR.PENS flag is set, XPOS, YPOS, Z1, Z2 are normally transmitted.

Therefore, using pen detection together with tag function eases the post-processing of the buffer, especially to determine which touchscreen converted values correspond to a period of time when the pen was in contact with the screen.

When the pen detection is disabled or the tag function is disabled, XPOS, YPOS, Z1, Z2 are normally transmitted without tag and no relationship can be found with pen status, thus post-processing may not be easy.

**Figure 65-24. Buffer Structure With and Without Pen Detection Enabled**




### 65.6.18 Fault Event

The ADC Controller internal fault output is directly connected to the PWM fault input. The fault event may be asserted depending on the configuration of ADC_EMR, ADC_CWR, ADC_LCMR and ADC_LCCWR and converted values.

Two types of comparison can trigger a comparison event (fault output pulse):

- The first comparison type is based on ADC_LCCWR settings, i.e., on all converted channels except the last one;
- The second comparison type is linked to the last channel.

As an example, overcurrent and temperature exceeding limits can trigger a fault to PWM.

When the comparison event occurs, the ADC fault output generates a pulse of one peripheral clock cycle to the PWM fault input. This fault line can be enabled or disabled within PWM. Should it be activated and asserted by the ADC Controller, the PWM outputs are immediately placed in a safe state (pure combinational path). Note that the ADC fault output connected to the PWM is not the COMPE bit. Thus the Fault mode (FMOD) within the PWM configuration must be FMOD = 1.

### 65.6.19 Register Write Protection

To prevent any single software error from corrupting ADC behavior, certain registers in the address space can be write-protected by setting the bit WPEN in the "ADC Write Protection Mode Register" (ADC_WPMR).

If a write access to the protected registers is detected, the WPVS flag in the "ADC Write Protection Status Register" (ADC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS flag is automatically reset by reading ADC_WPSR.

The following registers are write-protected when ADC_WPMR.WPEN is set:

- ADC Mode Register
- ADC Channel Sequence 1 Register
- ADC Channel Sequence 2 Register
- ADC Channel Enable Register
- ADC Channel Disable Register
- ADC Last Channel Trigger Mode Register
- ADC Last Channel Compare Window Register
- ADC Extended Mode Register
- ADC Compare Window Register
- ADC Channel Offset Register
- ADC Analog Control Register
- ADC_Touchscreen Mode Register
- ADC Trigger Register
- ADC Correction Values Register
- ADC Channel Error Correction Register
- ADC Touchscreen Correction Values Register

## 65.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | ADC_CR | 7:0 | | | | CMPRST | | TSCALIB | START | SWRST |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | ADC_MR | 7:0 | | FWUP | SLEEP | | TRGSEL[2:0] | | | |
| | | 15:8 | PRESCAL[7:0] | | | | | | | |
| | | 23:16 | ANACH | | | | STARTUP[3:0] | | | |
| | | 31:24 | USEQ | MAXSPEED | TRANSFER[1:0] | | TRACKTIM[3:0] | | | |
| 0x08 | ADC_SEQR1 | 7:0 | USCH2[3:0] | | | | USCH1[3:0] | | | |
| | | 15:8 | USCH4[3:0] | | | | USCH3[3:0] | | | |
| | | 23:16 | USCH6[3:0] | | | | USCH5[3:0] | | | |
| | | 31:24 | USCH8[3:0] | | | | USCH7[3:0] | | | |
| 0x0C | ADC_SEQR2 | 7:0 | | | | | USCHx[3:0] | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | ADC_CHER | 7:0 | CH7 | CH6 | CH5 | CH4 | CH3 | CH2 | CH1 | CH0 |
| | | 15:8 | | | | | CH11 | CH10 | CH9 | CH8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | ADC_CHDR | 7:0 | CH7 | CH6 | CH5 | CH4 | CH3 | CH2 | CH1 | CH0 |
| | | 15:8 | | | | | CH11 | CH10 | CH9 | CH8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | ADC_CHSR | 7:0 | CH7 | CH6 | CH5 | CH4 | CH3 | CH2 | CH1 | CH0 |
| | | 15:8 | | | | | CH11 | CH10 | CH9 | CH8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C ... 0x1F | Reserved | | | | | | | | | |
| 0x20 | ADC_LCDR | 7:0 | LDATA[7:0] | | | | | | | |
| | | 15:8 | LDATA[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | CHNBOSR[4:0] | | | | |
| 0x24 | ADC_IER | 7:0 | EOC7 | EOC6 | EOC5 | EOC4 | EOC3 | EOC2 | EOC1 | EOC0 |
| | | 15:8 | | | | | EOC11 | EOC10 | EOC9 | EOC8 |
| | | 23:16 | | PRDY | YRDY | XRDY | LCCHG | | | |
| | | 31:24 | | NOPEN | PEN | | | COMPE | GOVRE | DRDY |
| 0x28 | ADC_IDR | 7:0 | EOC7 | EOC6 | EOC5 | EOC4 | EOC3 | EOC2 | EOC1 | EOC0 |
| | | 15:8 | | | | | EOC11 | EOC10 | EOC9 | EOC8 |
| | | 23:16 | | PRDY | YRDY | XRDY | LCCHG | | | |
| | | 31:24 | | NOPEN | PEN | | | COMPE | GOVRE | DRDY |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x2C | ADC_IMR | 7:0 | EOC7 | EOC6 | EOC5 | EOC4 | EOC3 | EOC2 | EOC1 | EOC0 |
| | | 15:8 | | | | | EOC11 | EOC10 | EOC9 | EOC8 |
| | | 23:16 | | PRDY | YRDY | XRDY | LCCHG | | | |
| | | 31:24 | | NOPEN | PEN | | | COMPE | GOVRE | DRDY |
| 0x30 | ADC_ISR | 7:0 | EOC7 | EOC6 | EOC5 | EOC4 | EOC3 | EOC2 | EOC1 | EOC0 |
| | | 15:8 | | | | | EOC11 | EOC10 | EOC9 | EOC8 |
| | | 23:16 | | PRDY | YRDY | XRDY | LCCHG | | | |
| | | 31:24 | PENS | NOPEN | PEN | | | COMPE | GOVRE | DRDY |
| 0x34 | ADC_LCTMR | 7:0 | | | CMPMOD[1:0] | | | | | DUALTRIG |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x38 | ADC_LCCWR | 7:0 | LOWTHRES[7:0] | | | | | | | |
| | | 15:8 | | | | | LOWTHRES[11:8] | | | |
| | | 23:16 | HIGHTHRES[7:0] | | | | | | | |
| | | 31:24 | | | | | HIGHTHRES[11:8] | | | |
| 0x3C | ADC_OVER | 7:0 | OVRE7 | OVRE6 | OVRE5 | OVRE4 | OVRE3 | OVRE2 | OVRE1 | OVRE0 |
| | | 15:8 | | | | | OVRE11 | OVRE10 | OVRE9 | OVRE8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x40 | ADC_EMR | 7:0 | CMPSEL[3:0] | | | | | CMPTYPE | CMPMODE[1:0] | |
| | | 15:8 | | | CMPFILTER[1:0] | | | | CMPALL | |
| | | 23:16 | | | SRCCLK | ASTE | | | OSR[1:0] | |
| | | 31:24 | | | ADCMODE[1:0] | | | SIGNMODE[1:0] | | TAG |
| 0x44 | ADC_CWR | 7:0 | LOWTHRES[7:0] | | | | | | | |
| | | 15:8 | | | LOWTHRES[13:8] | | | | | |
| | | 23:16 | HIGHTHRES[7:0] | | | | | | | |
| | | 31:24 | | | HIGHTHRES[13:8] | | | | | |
| 0x48 ... 0x4B | Reserved | | | | | | | | | |
| 0x4C | ADC_COR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | DIFF7 | DIFF6 | DIFF5 | DIFF4 | DIFF3 | DIFF2 | DIFF1 | DIFF0 |
| | | 31:24 | | | | | DIFF11 | DIFF10 | DIFF9 | DIFF8 |
| 0x50 | ADC_CDR0 | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | | | DATA[13:8] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| ... | | | | | | | | | | |
| 0x7C | ADC_CDR11 | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | | | DATA[13:8] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x80 ... 0x93 | Reserved | | | | | | | | | |
| 0x94 | ADC_ACR | 7:0 | | | | | | | PENDETSENS[1:0] | |
| | | 15:8 | | | | | | | IBCTL[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x98 ... 0xAF | Reserved | | | | | | | | | |
| 0xB0 | ADC_TSMR | 7:0 | | | TSAV[1:0] | | | | TSMODE[1:0] | |
| | | 15:8 | | | | | TSFREQ[3:0] | | | |
| | | 23:16 | | NOTSDMA | | | TSSCTIM[3:0] | | | |
| | | 31:24 | PENDBC[3:0] | | | | | | | PENDET |
| 0xB4 | ADC_XPOSR | 7:0 | XPOS[7:0] | | | | | | | |
| | | 15:8 | | | | | XPOS[11:8] | | | |
| | | 23:16 | XSCALE[7:0] | | | | | | | |
| | | 31:24 | | | | | XSCALE[11:8] | | | |
| 0xB8 | ADC_YPOSR | 7:0 | YPOS[7:0] | | | | | | | |
| | | 15:8 | | | | | YPOS[11:8] | | | |
| | | 23:16 | YSCALE[7:0] | | | | | | | |
| | | 31:24 | | | | | YSCALE[11:8] | | | |
| 0xBC | ADC_PRESSR | 7:0 | Z1[7:0] | | | | | | | |
| | | 15:8 | | | | | Z1[11:8] | | | |
| | | 23:16 | Z2[7:0] | | | | | | | |
| | | 31:24 | | | | | Z2[11:8] | | | |
| 0xC0 | ADC_TRGR | 7:0 | | | | | | TRGMOD[2:0] | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | TRGPER[7:0] | | | | | | | |
| | | 31:24 | TRGPER[15:8] | | | | | | | |
| 0xC4 ... 0xD3 | Reserved | | | | | | | | | |
| 0xD4 | ADC_CVR | 7:0 | OFFSETCORR[7:0] | | | | | | | |
| | | 15:8 | OFFSETCORR[15:8] | | | | | | | |
| | | 23:16 | GAINCORR[7:0] | | | | | | | |
| | | 31:24 | GAINCORR[15:8] | | | | | | | |
| 0xD8 | ADC_CECR | 7:0 | ECORR7 | ECORR6 | ECORR5 | ECORR4 | ECORR3 | ECORR2 | ECORR1 | ECORR0 |
| | | 15:8 | | | | | ECORR11 | ECORR10 | ECORR9 | ECORR8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xDC | ADC_TSCVR | 7:0 | TSOFFSETCORR[7:0] | | | | | | | |
| | | 15:8 | TSOFFSETCORR[15:8] | | | | | | | |
| | | 23:16 | TSGAINCORR[7:0] | | | | | | | |
| | | 31:24 | TSGAINCORR[15:8] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xE0 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | ADC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | ADC_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |

### 65.7.1 ADC Control Register

**Name:** ADC_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | CMPRST | | TSCALIB | START | SWRST |
| Access | | | | W | | W | W | W |
| Reset | | | | – | | – | – | – |

**Bit 4 – CMPRST** Comparison Restart

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Stops the conversion result storage until the next comparison match. |

**Bit 2 – TSCALIB** Touchscreen Calibration
If conversion is in progress, the calibration sequence starts at the beginning of a new conversion sequence. If no conversion is in progress, the calibration sequence starts at the second conversion sequence located after the TSCALIB command (Sleep mode, waiting for a trigger event).

TSCALIB measurement sequence does not affect the Last Converted Data register (ADC_LCDR).

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Programs screen calibration (VDD/GND measurement) |

**Bit 1 – START** Start Conversion

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Begins analog-to-digital conversion. |

**Bit 0 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the ADC, simulating a hardware reset. |

### 65.7.2 ADC Mode Register

**Name:** ADC_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | USEQ | MAXSPEED | TRANSFER[1:0] | | TRACKTIM[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ANACH | | | | STARTUP[3:0] | | | |
| Access | R/W | | | | R/W | R/W | R/W | R/W |
| Reset | 0 | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PRESCAL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | FWUP | SLEEP | | TRGSEL[2:0] | | | |
| Access | | R/W | R/W | | R/W | R/W | R/W | |
| Reset | | 0 | 0 | | 0 | 0 | 0 | |

**Bit 31 – USEQ** User Sequence Enable

| Value | Name | Description |
|---|---|---|
| 0 | NUM_ORDER | Normal mode: The controller converts channels in a simple numeric order depending only on the channel index. |
| 1 | REG_ORDER | User Sequence mode: The sequence respects what is defined in ADC_SEQR1 and ADC_SEQR2 registers and can be used to convert the same channel several times. |

**Bit 30 – MAXSPEED** Maximum Sampling Rate Enable in Freerun Mode
This bit should always be set to 0.

**Bits 29:28 – TRANSFER[1:0]** Transfer Time
The TRANSFER field must be set to 2 to guarantee the optimal transfer time.

**Bits 27:24 – TRACKTIM[3:0]** Tracking Time

| Value | Name | Description |
|---|---|---|
| 0 | ADCCLK6 | The tracking time is 6 ADC clock cycles. |
| 1-14 | – | The tracking time is 6 ADC clock cycles. |
| 15 | ADCCLK7 | The tracking time is 7 ADC clock cycles. |

**Bit 23 – ANACH** Analog Change

| Value | Name | Description |
|---|---|---|
| 0 | NONE | No analog change on channel switching: DIFF0 is used for all channels. |
| 1 | ALLOWED | Allows different analog settings for each channel. See ADC_COR. |

**Bits 19:16 – STARTUP[3:0]** Startup Time

| Value | Name | Description |
|---|---|---|
| 0 | SUT0 | 0 periods of ADCCLK |
| 1 | SUT8 | 8 periods of ADCCLK |
| 2 | SUT16 | 16 periods of ADCCLK |
| 3 | SUT24 | 24 periods of ADCCLK |
| 4 | SUT64 | 64 periods of ADCCLK |
| 5 | SUT80 | 80 periods of ADCCLK |
| 6 | SUT96 | 96 periods of ADCCLK |
| 7 | SUT112 | 112 periods of ADCCLK |
| 8 | SUT512 | 512 periods of ADCCLK |
| 9 | SUT576 | 576 periods of ADCCLK |
| 10 | SUT640 | 640 periods of ADCCLK |
| 11 | SUT704 | 704 periods of ADCCLK |
| 12 | SUT768 | 768 periods of ADCCLK |
| 13 | SUT832 | 832 periods of ADCCLK |
| 14 | SUT896 | 896 periods of ADCCLK |
| 15 | SUT960 | 960 periods of ADCCLK |

**Bits 15:8 – PRESCAL[7:0]** Prescaler Rate Selection
PRESCAL = ($f_{peripheral\ clock}$ / (2 × $f_{ADCCLK}$)) – 1.

**Bit 6 – FWUP** Fast Wakeup

| Value | Name | Description |
|---|---|---|
| 0 | OFF | If SLEEP is 1, then both ADC core and reference voltage circuitry are OFF between conversions |
| 1 | ON | If SLEEP is 1, then Fast Wakeup Sleep mode: The voltage reference is ON between conversions and ADC core is OFF |

**Bit 5 – SLEEP** Sleep Mode

| Value | Name | Description |
|---|---|---|
| 0 | NORMAL | Normal Mode: The ADC core and reference voltage circuitry are kept ON between conversions. |
| 1 | SLEEP | Sleep Mode: The wakeup time can be modified by programming the FWUP bit. |

**Bits 3:1 – TRGSEL[2:0]** Trigger Selection
The trigger selection can be performed only if ADC_TRGR.TRGMOD = 1, 2 or 3.

| Value | Name | Description |
|---|---|---|
| 0 | ADC_TRIG0 | ADTRG |
| 1 | ADC_TRIG1 | TIOA0 |
| 2 | ADC_TRIG2 | TIOA1 |

| Value | Name | Description |
|---|---|---|
| 3 | ADC_TRIG3 | TIOA2 |
| 4 | ADC_TRIG4 | PWM event line 0 |
| 5 | ADC_TRIG5 | PWM event line 1 |
| 6 | ADC_TRIG6 | TIOA3 |
| 7 | ADC_TRIG7 | RTCOUT0 |

### 65.7.3 ADC Channel Sequence 1 Register

**Name:** ADC_SEQR1
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | USCH8[3:0] | | | | USCH7[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | USCH6[3:0] | | | | USCH5[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | USCH4[3:0] | | | | USCH3[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | USCH2[3:0] | | | | USCH1[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0:3, 4:7, 8:11, 12:15, 16:19, 20:23, 24:27, 28:31 – USCHx** User Sequence Number x
This register can be used only if the ADC_MR.USEQ field is set to ‘1’.

Any USCHx field is processed only if the ADC_CHSR.CHx-1 bit reads logical ‘1’, else any value written in USCHx does not add the corresponding channel in the conversion sequence.

Configuring the same value in different fields leads to multiple samples of the same channel during the conversion sequence. This can be done consecutively, or not, according to user needs.

Example: for each trigger event, to obtain the “CH3 CH1 CH0 CH4 CH4” conversion sequence, use the following settings:

ADC_SEQR1.USCH1=3, ADC_CHSR.CH0=1

ADC_SEQR1.USCH2=1, ADC_CHSR.CH1=1

ADC_SEQR1.USCH3=0, ADC_CHSR.CH2=1

ADC_SEQR1.USCH4=4, ADC_CHSR.CH3=1

ADC_SEQR1.USCH5=4, ADC_CHSR.CH4=1

### 65.7.4 ADC Channel Sequence 2 Register

**Name:** ADC_SEQR2
**Offset:** 0x0C
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | USCHx[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 3:0 – USCHx[3:0]** User Sequence Number x
This register can be used only if the ADC_MR.USEQ field is set to ‘1’.

Any USCHx field is processed only if the ADC_CHSR.CHx-1 bit reads logical ‘1’, else any value written in USCHx does not add the corresponding channel in the conversion sequence.

Configuring the same value in different fields leads to multiple samples of the same channel during the conversion sequence. This can be done consecutively, or not, according to user needs.

### 65.7.5 ADC Channel Enable Register

**Name:** ADC_CHER
**Offset:** 0x10
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CH11 | CH10 | CH9 | CH8 |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CH7 | CH6 | CH5 | CH4 | CH3 | CH2 | CH1 | CH0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – CHx** Channel x Enable
If ADC_MR.USEQ = 1, CHx corresponds to the enable of sequence number x+1 described in ADC_SEQR1 and ADC_SEQR2 (e.g. CH0 enables sequence number USCH1).

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding channel. |

### 65.7.6 ADC Channel Disable Register

**Name:** ADC_CHDR
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CH11 | CH10 | CH9 | CH8 |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CH7 | CH6 | CH5 | CH4 | CH3 | CH2 | CH1 | CH0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – CHx** Channel x Disable

**WARNING** If the corresponding channel is disabled during a conversion or if it is disabled and then reenabled during a conversion, its associated data and corresponding EOCx and GOVRE flags in ADC_ISR and OVREx flags in ADC_OVER are unpredictable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding channel. |

### 65.7.7 ADC Channel Status Register

**Name:** ADC_CHSR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CH11 | CH10 | CH9 | CH8 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CH7 | CH6 | CH5 | CH4 | CH3 | CH2 | CH1 | CH0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – CHx** Channel x Status

| Value | Description |
|---|---|
| 0 | The corresponding channel (or part of sequence, see ADC_SEQyR.USCHx field) is disabled. |
| 1 | The corresponding channel (or part of sequence, see ADC_SEQyR.USCHx field) is enabled.<br>Note: As an example, when ADC_MR.USEQ=1 and a 1 is written to CH2, the channel defined in ADC_SEQ0R.USCH3 is part of the sequence of conversions. |

### 65.7.8 ADC Last Converted Data Register

**Name:** ADC_LCDR
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | CHNBOSR[4:0] | | | | |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LDATA[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LDATA[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 28:24 – CHNBOSR[4:0]** Channel Number in Oversampling Mode
Indicates the last converted channel when the ADC_EMR.TAG bit is set and the ADC_EMR0.OSR field is not equal to 0. If the ADC_EMR.TAG bit is not set, CHNBOSR = 0.

**Bits 15:0 – LDATA[15:0]** Last Data Converted
The analog-to-digital conversion data is placed into this register at the end of a conversion and remains until a new conversion is completed.

If OSR = 0 and TAG = 1 in ADC_EMR, the 4 MSBs of LDATA carry the channel number to obtain a packed system memory buffer made of 1 converted data stored in a halfword (16-bit) instead of 1 converted data in a 32-bit word, thus dividing by 2 the size of the memory buffer.

### 65.7.9 ADC Interrupt Enable Register

**Name:** ADC_IER
**Offset:** 0x24
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | NOPEN | PEN | | | COMPE | GOVRE | DRDY |
| Access | | W | W | | | W | W | W |
| Reset | | – | – | | | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | PRDY | YRDY | XRDY | LCCHG | | | |
| Access | | W | W | W | W | | | |
| Reset | | – | – | – | – | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOC11 | EOC10 | EOC9 | EOC8 |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EOC7 | EOC6 | EOC5 | EOC4 | EOC3 | EOC2 | EOC1 | EOC0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 30 – NOPEN** No Pen Contact Interrupt Enable

**Bit 29 – PEN** Pen Contact Interrupt Enable

**Bit 26 – COMPE** Comparison Event Interrupt Enable

**Bit 25 – GOVRE** General Overrun Error Interrupt Enable

**Bit 24 – DRDY** Data Ready Interrupt Enable

**Bit 22 – PRDY** Touchscreen Measure Pressure Ready Interrupt Enable

**Bit 21 – YRDY** Touchscreen Measure YPOS Ready Interrupt Enable

**Bit 20 – XRDY** Touchscreen Measure XPOS Ready Interrupt Enable

**Bit 19 – LCCHG** Last Channel Change Interrupt Enable

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – EOCx** End of Conversion Interrupt Enable x

### 65.7.10 ADC Interrupt Disable Register

**Name:** ADC_IDR
**Offset:** 0x28
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | NOPEN | PEN | | | COMPE | GOVRE | DRDY |
| Access | | W | W | | | W | W | W |
| Reset | | – | – | | | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | PRDY | YRDY | XRDY | LCCHG | | | |
| Access | | W | W | W | W | | | |
| Reset | | – | – | – | – | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOC11 | EOC10 | EOC9 | EOC8 |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EOC7 | EOC6 | EOC5 | EOC4 | EOC3 | EOC2 | EOC1 | EOC0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 30 – NOPEN** No Pen Contact Interrupt Disable

**Bit 29 – PEN** Pen Contact Interrupt Disable

**Bit 26 – COMPE** Comparison Event Interrupt Disable

**Bit 25 – GOVRE** General Overrun Error Interrupt Disable

**Bit 24 – DRDY** Data Ready Interrupt Disable

**Bit 22 – PRDY** Touchscreen Measure Pressure Ready Interrupt Disable

**Bit 21 – YRDY** Touchscreen Measure YPOS Ready Interrupt Disable

**Bit 20 – XRDY** Touchscreen Measure XPOS Ready Interrupt Disable

**Bit 19 – LCCHG** Last Channel Change Interrupt Disable

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – EOCx** End of Conversion Interrupt Disable x

### 65.7.11 ADC Interrupt Mask Register

**Name:** ADC_IMR
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | NOPEN | PEN | | | COMPE | GOVRE | DRDY |
| Access | | R | R | | | R | R | R |
| Reset | | 0 | 0 | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | PRDY | YRDY | XRDY | LCCHG | | | |
| Access | | R | R | R | R | | | |
| Reset | | 0 | 0 | 0 | 0 | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOC11 | EOC10 | EOC9 | EOC8 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EOC7 | EOC6 | EOC5 | EOC4 | EOC3 | EOC2 | EOC1 | EOC0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 30 – NOPEN** No Pen Contact Interrupt Mask

**Bit 29 – PEN** Pen Contact Interrupt Mask

**Bit 26 – COMPE** Comparison Event Interrupt Mask

**Bit 25 – GOVRE** General Overrun Error Interrupt Mask

**Bit 24 – DRDY** Data Ready Interrupt Mask

**Bit 22 – PRDY** Touchscreen Measure Pressure Ready Interrupt Mask

**Bit 21 – YRDY** Touchscreen Measure YPOS Ready Interrupt Mask

**Bit 20 – XRDY** Touchscreen Measure XPOS Ready Interrupt Mask

**Bit 19 – LCCHG** Last Channel Change Interrupt Disable

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – EOCx** End of Conversion Interrupt Mask x

### 65.7.12 ADC Interrupt Status Register

**Name:** ADC_ISR
**Offset:** 0x30
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PENS | NOPEN | PEN | | | COMPE | GOVRE | DRDY |
| Access | R | R | R | | | R | R | R |
| Reset | 0 | 0 | 0 | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | PRDY | YRDY | XRDY | LCCHG | | | |
| Access | | R | R | R | R | | | |
| Reset | | 0 | 0 | 0 | 0 | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOC11 | EOC10 | EOC9 | EOC8 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EOC7 | EOC6 | EOC5 | EOC4 | EOC3 | EOC2 | EOC1 | EOC0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – PENS** Pen Detect Status

| Value | Description |
|---|---|
| 0 | The pen does not press the screen. |
| 1 | The pen presses the screen.<br><br>Note: PENS is not a source of interruption. |

**Bit 30 – NOPEN** No Pen Contact (cleared on read)

| Value | Description |
|---|---|
| 0 | No loss of pen contact since the last read of ADC_ISR. |
| 1 | At least one loss of pen contact since the last read of ADC_ISR. |

**Bit 29 – PEN** Pen contact (cleared on read)

| Value | Description |
|---|---|
| 0 | No pen contact since the last read of ADC_ISR. |
| 1 | At least one pen contact since the last read of ADC_ISR. |

**Bit 26 – COMPE** Comparison Event (cleared on read)

| Value | Description |
|---|---|
| 0 | No comparison event since the last read of ADC_ISR. |
| 1 | At least one comparison event (defined in ADC_EMR and ADC_CWR) has occurred since the last read of ADC_ISR. |

**Bit 25 – GOVRE** General Overrun Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No general overrun error occurred since the last read of ADC_ISR. |
| 1 | At least one general overrun error has occurred since the last read of ADC_ISR. |

**Bit 24 – DRDY** Data Ready (automatically set / cleared)

| Value | Description |
|---|---|
| 0 | No data has been converted since the last read of ADC_LCDR. |
| 1 | At least one data has been converted and is available in ADC_LCDR. |

**Bit 22 – PRDY** Touchscreen Pressure Measure Ready (cleared on read)

| Value | Description |
|---|---|
| 0 | No measure has been performed since the last read of ADC_PRESSR. |
| 1 | At least one measure has been performed since the last read of ADC_ISR. |

**Bit 21 – YRDY** Touchscreen YPOS Measure Ready (cleared on read)

| Value | Description |
|---|---|
| 0 | No measure has been performed since the last read of ADC_YPOSR. |
| 1 | At least one measure has been performed since the last read of ADC_ISR. |

**Bit 20 – XRDY** Touchscreen XPOS Measure Ready (cleared on read)

| Value | Description |
|---|---|
| 0 | No measure has been performed since the last read of ADC_XPOSR. |
| 1 | At least one measure has been performed since the last read of ADC_ISR. |

**Bit 19 – LCCHG** Last Channel Change (cleared on read)

| Value | Description |
|---|---|
| 0 | There is no comparison match (defined in the Last Channel Compare Window register (ADC_LCCWR) since the last read of ADC_ISR. |
| 1 | The converted value reported on ADC_CDR11 has changed since the last read of ADC_ISR, according to what is defined in ADC_LCTMR and ADC_LCCWR. |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – EOCx** End of Conversion x (automatically set / cleared)

| Value | Description |
|---|---|
| 0 | The corresponding analog channel is disabled, or the conversion is not finished. This flag is cleared when reading the corresponding ADC_CDRx registers. |
| 1 | The corresponding analog channel is enabled and conversion is complete. |

### 65.7.13 ADC Last Channel Trigger Mode Register

**Name:** ADC_LCTMR
**Offset:** 0x34
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | CMPMOD[1:0] | | | | | DUALTRIG |
| Access | | | R/W | R/W | | | | R/W |
| Reset | | | 0 | 0 | | | | 0 |

**Bits 5:4 – CMPMOD[1:0]** Last Channel Comparison Mode

| Value | Name | Description |
|---|---|---|
| 0 | LOW | Generates the ADC_ISR.LCCHG flag when the converted data is lower than the low threshold of the window. |
| 1 | HIGH | Generates the ADC_ISR.LCCHG flag when the converted data is higher than the high threshold of the window. |
| 2 | IN | Generates the ADC_ISR.LCCHG flag when the converted data is in the comparison window. |
| 3 | OUT | Generates the ADC_ISR.LCCHG flag when the converted data is out of the comparison window. |

**Bit 0 – DUALTRIG** Dual Trigger ON

| Value | Description |
|---|---|
| 0 | All channels are triggered by event defined by ADC_MR.TRGSEL. |
| 1 | Last channel (higher index) trigger period is defined by RTC_MR.OUT1. |

### 65.7.14 ADC Last Channel Compare Window Register

**Name:** ADC_LCCWR
**Offset:** 0x38
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | HIGHTHRES[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | HIGHTHRES[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | LOWTHRES[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LOWTHRES[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27:16 – HIGHTHRES[11:0]** High Threshold
High threshold associated to compare settings of ADC_LCTMR.

**Bits 11:0 – LOWTHRES[11:0]** Low Threshold
Low threshold associated to compare settings of ADC_LCTMR.

### 65.7.15 ADC Overrun Status Register

**Name:** ADC_OVER
**Offset:** 0x3C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRE11 | OVRE10 | OVRE9 | OVRE8 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVRE7 | OVRE6 | OVRE5 | OVRE4 | OVRE3 | OVRE2 | OVRE1 | OVRE0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – OVREx** Overrun Error x

| Value | Description |
|---|---|
| 0 | No overrun error on the corresponding channel since the last read of ADC_OVER. |
| 1 | An overrun error has occurred on the corresponding channel since the last read of ADC_OVER. |

### 65.7.16 ADC Extended Mode Register

**Name:** ADC_EMR
**Offset:** 0x40
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | ADCMODE[1:0] | | | SIGNMODE[1:0] | | TAG |
| Access | | | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | 0 | 0 | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | SRCCLK | ASTE | | | OSR[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | CMPFILTER[1:0] | | | | CMPALL | |
| Access | | | R/W | R/W | | | R/W | |
| Reset | | | 0 | 0 | | | 0 | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMPSEL[3:0] | | | | | CMPTYPE | CMPMODE[1:0] | |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bits 29:28 – ADCMODE[1:0]** ADC Running Mode
See 65.6.14 Automatic Error Correction for details on ADC running mode.

| Value | Name | Description |
|---|---|---|
| 0 | NORMAL | Normal mode of operation. |
| 1 | OFFSET_ERROR | Offset Error mode to measure the offset error. See table ADC Running Modes. |
| 2 | GAIN_ERROR_HIGH | Gain Error mode to measure the gain error. See table ADC Running Modes. |
| 3 | GAIN_ERROR_LOW | Gain Error mode to measure the gain error. See table ADC Running Modes. |

**Bits 26:25 – SIGNMODE[1:0]** Sign Mode
If conversion results are signed and resolution is below 16 bits, the sign is extended up to the bit 15 (for example, 0xF43 for 12-bit resolution will be read as 0xFF43 and 0x467 will be read as 0x0467). See 65.6.6 Conversion Results Format.

| Value | Name | Description |
|---|---|---|
| 0 | SE_UNSG_DF_SIGN | Single-Ended channels: Unsigned conversions.<br>Differential channels: Signed conversions. |

| Value | Name | Description |
|---|---|---|
| 1 | SE_SIGN_DF_UNSG | Single-Ended channels: Signed conversions.<br>Differential channels: Unsigned conversions. |
| 2 | ALL_UNSIGNED | All channels: Unsigned conversions. |
| 3 | ALL_SIGNED | All channels: Signed conversions. |

**Bit 24 – TAG** Tag of ADC_LCDR

| Value | Description |
|---|---|
| 0 | Sets ADC_LCDR.CHNB field to zero. |
| 1 | Appends the channel number to the conversion result in ADC_LCDR. |

**Bit 21 – SRCCLK** External Clock Selection
0 (PERIPH_CLK): The peripheral clock is the source for the ADC prescaler.

1 (GCLK): GCLK is the source clock for the ADC prescaler, thus the ADC clock can be independent of the core/peripheral clock.

**Bit 20 – ASTE** Averaging on Single Trigger Event

| Value | Name | Description |
|---|---|---|
| 0 | MULTI_TRIG_AVERAGE | The average requests several trigger events. |
| 1 | SINGLE_TRIG_AVERAGE | The average requests only one trigger event. |

**Bits 17:16 – OSR[1:0]** Over Sampling Rate

| Value | Name | Description |
|---|---|---|
| 0 | NO_AVERAGE | No averaging. ADC sample rate is maximum. |
| 1 | OSR4 | 1-bit enhanced resolution by averaging. ADC sample rate divided by 4. |
| 2 | OSR16 | 2-bit enhanced resolution by averaging. ADC sample rate divided by 16. |
| 3 | OSR64 | 3-bit enhanced resolution by averaging. ADC sample rate divided by 64. |
| 4 | OSR256 | 4-bit enhanced resolution by averaging. ADC sample rate divided by 256. |

**Bits 13:12 – CMPFILTER[1:0]** Compare Event Filtering
Number of consecutive compare events necessary to raise the flag = CMPFILTER+1

When programmed to 0, the flag rises as soon as an event occurs.

See 65.6.9 Comparison Window when using the filtering option (CMPFILTER > 0).

**Bit 9 – CMPALL** Compare All Channels

| Value | Description |
|---|---|
| 0 | Only channel indicated in CMPSEL field is compared. |
| 1 | All channels are compared. |

**Bits 7:4 – CMPSEL[3:0]** Comparison Selected Channel
If CMPALL = 0: CMPSEL indicates which channel has to be compared.

If CMPALL = 1: No effect.

**Bit 2 – CMPTYPE** Comparison Type

| Value | Name | Description |
|---|---|---|
| 0 | FLAG_ONLY | Any conversion is performed and comparison function drives the ADC_ISR.COMPE flag. |
| 1 | START_CONDITION | Comparison conditions must be met to start the storage of all conversions until the ADC_CR.CMPRST bit is set. |

**Bits 1:0 – CMPMODE[1:0]** Comparison Mode

| Value | Name | Description |
|---|---|---|
| 0 | LOW | When the converted data is lower than the low threshold of the window, generates the ADC_ISR.COMPE flag or, in Partial Wakeup mode, defines the conditions to exit system from Wait mode. |
| 1 | HIGH | When the converted data is higher than the high threshold of the window, generates the ADC_ISR.COMPE flag or, in Partial Wakeup mode, defines the conditions to exit system from Wait mode. |
| 2 | IN | When the converted data is in the comparison window, generates the ADC_ISR.COMPE flag or, in Partial Wakeup mode, defines the conditions to exit system from Wait mode. |
| 3 | OUT | When the converted data is out of the comparison window, generates the ADC_ISR.COMPE flag or, in Partial Wakeup mode, defines the conditions to exit system from Wait mode. |

© 2018 Microchip Technology Inc.

### 65.7.17 ADC Compare Window Register

**Name:** ADC_CWR
**Offset:** 0x44
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | HIGHTHRES[13:8] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | HIGHTHRES[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | LOWTHRES[13:8] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LOWTHRES[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:16 – HIGHTHRES[13:0]** High Threshold
High threshold associated to compare settings of ADC_EMR.

**Bits 13:0 – LOWTHRES[13:0]** Low Threshold
Low threshold associated to compare settings of ADC_EMR.

### 65.7.18 Channel Offset Register

**Name:** ADC_COR
**Offset:** 0x4C
**Reset:** 0
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | DIFF11 | DIFF10 | DIFF9 | DIFF8 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DIFF7 | DIFF6 | DIFF5 | DIFF4 | DIFF3 | DIFF2 | DIFF1 | DIFF0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27 – DIFFx** Differential Inputs for Channel x

| Value | Description |
|---|---|
| 0 | Corresponding channel is set in Single-ended mode. |
| 1 | Corresponding channel is set in Differential mode. |

### 65.7.19 ADC Channel Data Register

**Name:** ADC_CDRx
**Offset:** 0x50 + x*0x04 [x=0..11]
**Reset:** 0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | DATA[13:8] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 13:0 – DATA[13:0]** Converted Data
The analog-to-digital conversion data is placed into this register at the end of a conversion and remains until a new conversion is completed. ADC_CDRx is only loaded if the corresponding analog channel is enabled.

### 65.7.20 ADC Analog Control Register

**Name:** ADC_ACR
**Offset:** 0x94
**Reset:** 0x00000101
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

By default, bits 12 and 13 are set to 1 and 0, respectively, and must not be modified.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | IBCTL[1:0] | |
| Reset | | | | | | | 0 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | PENDETSENS[1:0] | |
| Reset | | | | | | | 0 | 1 |

**Bits 9:8 – IBCTL[1:0]** ADC Bias Current Control
Adapts performance versus power consumption. Refer to the “Electrical Characteristics” section for further details.

**Bits 1:0 – PENDETSENS[1:0]** Pen Detection Sensitivity
Modifies the pen detection input pull-up resistor value. Refer to the “Electrical Characteristics” section for further details.

### 65.7.21 ADC Touchscreen Mode Register

**Name:** ADC_TSMR
**Offset:** 0xB0
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PENDBC[3:0] | | | | | | | PENDET |
| Access | R/W | R/W | R/W | R/W | | | | R/W |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | NOTSDMA | | | TSSCTIM[3:0] | | | |
| Access | | R/W | | | R/W | R/W | R/W | R/W |
| Reset | | 0 | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TSFREQ[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TSAV[1:0] | | | | TSMODE[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bits 31:28 – PENDBC[3:0]** Pen Detect Debouncing Period
Debouncing period = $2^{PENDBC}$ ADCCLK periods.

**Bit 24 – PENDET** Pen Contact Detection Enable
When PENDET = 1, XPOS, YPOS, Z1, Z2 values of ADC_XPOSR, ADC_YPOSR, ADC_PRESSR are automatically cleared when ADC_ISR.PENS = 0.

| Value | Description |
|---|---|
| 0 | Pen contact detection disabled. |
| 1 | Pen contact detection enabled. |

**Bit 22 – NOTSDMA** No TouchScreen DMA

| Value | Description |
|---|---|
| 0 | XPOS, YPOS, Z1, Z2 are transmitted in ADC_LCDR. |
| 1 | XPOS, YPOS, Z1, Z2 are never transmitted in ADC_LCDR, therefore the buffer does not contains touchscreen values. |

**Bits 19:16 – TSSCTIM[3:0]** Touchscreen Switches Closure Time
Defines closure time of analog switches necessary to establish the measurement conditions.

The closure time is:

Switch Closure Time = (TSSCTIM × 4) ADCCLK periods.

**Bits 11:8 – TSFREQ[3:0]** Touchscreen Frequency
Defines the touchscreen frequency compared to the trigger frequency.

TSFREQ must be greater or equal to TSAV.

The touchscreen frequency is:

Touchscreen Frequency = Trigger Frequency / $2^{TSFREQ}$

**Bits 5:4 – TSAV[1:0]** Touchscreen Average

| Value | Name | Description |
|---|---|---|
| 0 | NO_FILTER | No Filtering. Only one ADC conversion per measure |
| 1 | AVG2CONV | Averages 2 ADC conversions |
| 2 | AVG4CONV | Averages 4 ADC conversions |
| 3 | AVG8CONV | Averages 8 ADC conversions |

**Bits 1:0 – TSMODE[1:0]** Touchscreen Mode
When TSMOD equals 01 or 10 (i.e., 4-wire mode), channels 0, 1, 2 and 3 must not be used for classic ADC conversions. When TSMOD equals 11 (i.e., 5-wire mode), channels 0, 1, 2, 3, and 4 must not be used.

| Value | Name | Description |
|---|---|---|
| 0 | NONE | No Touchscreen |
| 1 | 4_WIRE_NO_PM | 4-wire Touchscreen without pressure measurement |
| 2 | 4_WIRE | 4-wire Touchscreen with pressure measurement |
| 3 | 5_WIRE | 5-wire Touchscreen |

### 65.7.22 ADC Touchscreen X Position Register

**Name:** ADC_XPOSR
**Offset:** 0xB4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | XSCALE[11:8] | | | |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | XSCALE[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | XPOS[11:8] | | | |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XPOS[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27:16 – XSCALE[11:0]** Scale of XPOS
Indicates the max value that XPOS can reach. This value should be close to $2^{12}$.

**Bits 11:0 – XPOS[11:0]** X Position
The position measured is stored here. If XPOS = 0 or XPOS = XSIZE, the pen is on the border.

When pen detection is enabled (ADC_TSMR.PENDET set to ‘1’), XPOS is tied to 0 while there is no detection of contact on the touchscreen (i.e., when the ADC_ISR.PENS bit is cleared).

### 65.7.23 ADC Touchscreen Y Position Register

**Name:** ADC_YPOSR
**Offset:** 0xB8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | YSCALE[11:8] | | | |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | YSCALE[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | YPOS[11:8] | | | |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | YPOS[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27:16 – YSCALE[11:0]** Scale of YPOS
Indicates the max value that YPOS can reach. This value should be close to $2^{12}$.

**Bits 11:0 – YPOS[11:0]** Y Position
The position measured is stored here. If YPOS = 0 or YPOS = YSIZE, the pen is on the border.

When pen detection is enabled (ADC_TSMR.PENDET set to ‘1’), YPOS is tied to 0 while there is no detection of contact on the touchscreen (i.e., when the ADC_ISR.PENS bit is cleared).

### 65.7.24 ADC Touchscreen Pressure Register

**Name:** ADC_PRESSR
**Offset:** 0xBC
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | Z2[11:8] | | | |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | Z2[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | Z1[11:8] | | | |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | Z1[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27:16 – Z2[11:0]** Data of Z2 Measurement
Data Z2 necessary to calculate pen pressure.

When pen detection is enabled (ADC_TSMR.PENDET set to ‘1’), Z2 is tied to 0 while there is no detection of contact on the touchscreen (i.e., when the ADC_ISR.PENS bit is cleared).

**Note:** These two values are unavailable if ADC_TSMR.TSMODE is not set to 2.

**Bits 11:0 – Z1[11:0]** Data of Z1 Measurement
Data Z1 necessary to calculate pen pressure.

When pen detection is enabled (ADC_TSMR.PENDET set to ‘1’), Z1 is tied to 0 while there is no detection of contact on the touchscreen (i.e., when the ADC_ISR.PENS bit is cleared).

### 65.7.25 ADC Trigger Register

**Name:** ADC_TRGR
**Offset:** 0xC0
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TRGPER[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TRGPER[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | TRGMOD[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 31:16 – TRGPER[15:0]** Trigger Period
Effective only if TRGMOD defines a periodic trigger.

Defines the periodic trigger period, with the following equation:

Trigger Period = (TRGPER + 1) / ADCCLK

The minimum time between two consecutive trigger events must be strictly greater than the duration time of the longest conversion sequence depending on the configuration of registers ADC_MR, ADC_CHSR, ADC_SEQRx, ADC_TSMR.

When TRGMOD is set to pen detect trigger (i.e., 100) and averaging is used (i.e., field TSAV ≠ 0 in ADC_TSMR) only one measure is performed. Thus, XRDY, YRDY, PRDY, DRDY will not rise on pen contact trigger. To achieve measurement, several triggers must be provided either by software or by setting the TRGMOD on continuous trigger (i.e., 110) until flags rise.

**Bits 2:0 – TRGMOD[2:0]** Trigger Mode

| Value | Name | Description |
|---|---|---|
| 0 | NO_TRIGGER | No trigger, only software trigger can start conversions |
| 1 | EXT_TRIG_RISE | Rising edge of the selected trigger event, defined in ADC_MR.TRGSEL |
| 2 | EXT_TRIG_FALL | Falling edge of the selected trigger event |
| 3 | EXT_TRIG_ANY | Any edge of the selected trigger event |

| Value | Name | Description |
|---|---|---|
| 4 | PEN_TRIG | Pen Detect Trigger (shall be selected only if PENDET is set and TSMODE > 0) |
| 5 | PERIOD_TRIG | ADC internal periodic trigger (see field TRGPER) |
| 6 | CONTINUOUS | Continuous mode |

### 65.7.26 ADC Correction Values Register

**Name:** ADC_CVR
**Offset:** 0xD4
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | GAINCORR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GAINCORR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | OFFSETCORR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OFFSETCORR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – GAINCORR[15:0]** Gain Correction
Gain correction to apply on converted data. Only bits 0 to 13 are relevant (other bits are ignored and read as 0).

**Bits 15:0 – OFFSETCORR[15:0]** Offset Correction
Offset correction to apply on converted data. The offset is signed (2's complement), only bits 0 to 11 are relevant (other bits are ignored and read as 0).

### 65.7.27 ADC Channel Error Correction Register

**Name:** ADC_CECR
**Offset:** 0xD8
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | ECORR11 | ECORR10 | ECORR9 | ECORR8 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ECORR7 | ECORR6 | ECORR5 | ECORR4 | ECORR3 | ECORR2 | ECORR1 | ECORR0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – ECORRx** Error Correction Enable for channel x

| Value | Description |
|---|---|
| 0 | Automatic error correction is disabled for channel x. |
| 1 | Automatic error correction is enabled for channel x. |

### 65.7.28 ADC Touchscreen Correction Values Register

**Name:** ADC_TSCVR
**Offset:** 0xDC
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TSGAINCORR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TSGAINCORR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TSOFFSETCORR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TSOFFSETCORR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – TSGAINCORR[15:0]** Touchscreen Gain Correction
Gain correction to apply on converted data for the touchscreen channels. Only bits 0 to 13 are relevant (other bits are ignored and read as 0).

**Bits 15:0 – TSOFFSETCORR[15:0]** Touchscreen Offset Correction
Offset correction to apply on converted data for the touchscreen channels. The offset is signed (2's complement), only bits 0 to 11 are relevant (other bits are ignored and read as 0).

### 65.7.29 ADC Write Protection Mode Register

**Name:** ADC_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x414443 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit.<br>Always reads as 0 |

**Bit 0 – WPEN** Write Protection Enable
See 65.6.19 Register Write Protection for the list of write-protected registers.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY value corresponds to 0x414443 (“ADC” in ASCII). |
| 1 | Enables the write protection if WPKEY value corresponds to 0x414443 (“ADC” in ASCII). |

### 65.7.30 ADC Write Protection Status Register

**Name:** ADC_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of ADC_WPSR. |
| 1 | A write protection violation has occurred since the last read of ADC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 66. Electrical Characteristics

## 66.1 Absolute Maximum Ratings

**Table 66-1. Absolute Maximum Ratings***

| | | |
|---|---|---|
| **Storage Temperature** | -60°C to + 150°C | **Note:** Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability. |
| **Voltage on Input Pins with Respect to Ground** | -0.3V to +4.0V | |
| **Maximum Voltage on Power Input Pins with Respect to Ground** | | |
| VDDCORE, VDDPLLA, VDDUTMIC and VDDHSIC | 1.5V | |
| VDDIODDR | 2.0V | |
| VDDBU | 4.0V | |
| VDDIOPx, VDDUTMII, VDDISC, VDDSDMMC, VDDOSC, VDDANA, VDDAUDIOPLL | 4.0V | |
| VDDFUSE | 3.0V | |
| **Total DC Output Current on all I/O lines** | 350 mA | |

## 66.2 DC Characteristics

The following characteristics are applicable to the ambient operating temperature range $T_A$ = -40°C to +105°C for -CN devices, and -40°C to +85°C for -CU devices, unless otherwise specified.

**Table 66-2. Recommended Thermal Operating Conditions**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $T_A$ | Ambient Temperature | – | -40 | – | +105 | °C |
| $T_J$ | Junction Temperature | – | -40 | – | +125 | °C |
| $R_{thJA}$ | Junction-to-ambient thermal resistance | LFBGA289 | – | 34.8 | – | °C/W |
| | | TFBGA256 | – | 27.4 | – | |
| | | TFBGA196 | – | 44.6 | – | |
| $R_{thJC}$ | Junction-to-case thermal resistance | LFBGA289 | – | 10.9 | – | °C/W |
| | | TFBGA256 | – | 9.3 | – | |
| | | TFBGA196 | – | 11.2 | – | |
| $P_D$ | Allowable Power Dissipation | At $T_A$ = 85°C, LFBGA289 | – | – | 1149 | mW |
| | | At $T_A$ = 105°C, LFBGA289 | – | – | 575 | mW |
| $P_D$ | Allowable Power Dissipation | At $T_A$ = 85°C, TFBGA256 | – | – | 1460 | mW |
| | | At $T_A$ = 105°C, TFBGA256 | – | – | 730 | mW |

.....continued

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $P_D$ | Allowable Power Dissipation | At $T_A$ = 85°C, TFBGA196 | – | – | 897 | mW |
| | | At $T_A$ = 105°C, TFBGA196 | – | – | 448 | mW |
| $V_{DDCORE}$ | DC Supply Core | – | 1.1 | 1.2 | 1.32 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 20 MHz | – | – | 15 | mV |
| | Slope | – | 1.3 | – | – | V/ms |
| $V_{DDBU}$ | DC Supply I/Os, Backup | Must be established first | 1.65 | | 3.6 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 10 MHz | – | – | 30 | mV |
| | Slope | – | 2.4 | – | – | V/ms |
| $V_{DDANA}$ | DC Supply I/Os, Backup | The ADC is not functional below 2.0V | 1.65 | | 3.6 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 20 MHz | – | – | 20 | mV |
| | Slope | – | 2.4 | – | – | V/ms |
| $V_{DDIOP0}$ | DC Supply LCD I/Os | | 1.65 | – | 3.6 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 20 MHz | – | – | 20 | mV |
| $V_{DDIOP1}$ | DC Supply Peripheral I/Os | Peripheral I/O Lines | 1.65 | – | 3.6 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 20 MHz | – | – | 20 | mV |
| $V_{DDIOP2}$ | DC Supply Peripheral I/Os | Peripheral I/O Lines | 1.65 | – | 3.6 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 20 MHz | – | – | 20 | mV |
| $V_{DDPLLA}$ | PLL A and Main Oscillator Supply | – | 1.1 | 1.2 | 1.32 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 10 MHz | – | – | 20 | mV |
| | | rms value > 10 MHz | – | – | 20 | |
| $V_{DDUTMIC}$ | DC Supply UDPHS and UHPHS UTMI+ Core | – | 1.1 | 1.2 | 1.32 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 10 MHz | – | – | 20 | mV |
| $V_{DDUTMII}$ | DC Supply UDPHS and UHPHS UTMI+ Interface | – | 3.0 | 3.3 | 3.6 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 10 MHz | – | – | 20 | mV |
| $V_{DDHSIC}$ | DC Supply HSIC Phy | – | 1.1 | 1.2 | 1.3 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 10 MHz | – | – | 20 | mV |
| $V_{DDAUDIOPLL}$ | DC Supply AUDIO PLL | – | 3.0 | 3.3 | 3.6 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 10 MHz | – | – | 20 | mV |
| $V_{DDFUSE}$ | DC Supply Fuse Box | For fuse programming only | 2.25 | 2.5 | 2.75 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 10 MHz | – | – | 20 | mV |
| $V_{DDSDMMC}$ | DC Supply Peripheral I/Os | SDMMC I/Os Lines | 1.65 | – | 3.6 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 10 MHz | – | – | 20 | mV |
| $V_{DDISC}$ | DC Supply Peripheral I/Os | ISC I/Os Lines | 1.65 | – | 3.6 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 10 MHz | – | – | 20 | mV |

.....continued

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{DDOSC}$ | DC Supply Oscillator | – | 1.65 | – | 3.6 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 10 MHz | – | – | 15 | mV |
| $V_{DDIODDR}$ | DC Supply SDRAM I/Os | - LPDDR1-DDR2 Interface I/O lines | 1.7 | 1.8 | 1.9 | V |
| | | - LPDDR2-LPDDR3 Interface I/O lines | 1.14 | 1.2 | 1.30 | |
| | | - DDR3L Interface I/O lines | 1.29 | 1.35 | 1.45 | |
| | | - DDR3 Interface I/O lines | 1.43 | 1.5 | 1.57 | |
| $V_{IL}$ | Low-level Input Voltage(2) | VDDIO in 3.3V range | -0.3 | – | 0.8 | V |
| | | VDDIO in 1.8V range | -0.3 | – | 0.3 x VDDIO | |
| $V_{IH}$ | High-level Input Voltage(2) | VDDIO in 3.3V range | 2 | – | VDDIO +0.3 | V |
| | | VDDIO in 1.8V range | 0.7 x VDDIO | – | VDDIO +0.3 | |
| $V_{OL}$ | Low-level Output Voltage | $I_{OL}$ min | – | – | 0.41 | V |
| $V_{OH}$ | High-level Output Voltage | $I_{OH}$ max | VDDIO - 0.4 | – | – | V |
| $V_{hys}$ | Schmitt Trigger Hysteresis | All PIO lines, VDDIOx in 3.3V range | 200 | – | – | mV |
| | | All PIO lines, VDDIOx in 1.8V range | 150 | – | – | |
| $I_{OL}$ | Low-level Output Current with $V_{OL}$ = 0.41V | All GPIO_x, 1.8V: Low | -1 | – | – | mA |
| | | All GPIO_x, 1.8V: Medium | -10 | – | – | |
| | | All GPIO_x, 1.8V: High | -18 | – | – | |
| $I_{OH}$ | High-level Output Current with $V_{OH}$ = VDDIO - 0.4V | All GPIO_x, 1.8V: Low | – | – | 1 | mA |
| | | All GPIO_x, 1.8V: Medium | – | – | 10 | |
| | | All GPIO_x, 1.8V: High | – | – | 18 | |
| $I_{OL}$ | Low-level Output Current with $V_{OL}$ = 0.41V | All GPIO_x, 3.3V: Low | -2 | – | – | mA |
| | | All GPIO_x, 3.3V: Medium | -20 | – | – | |
| | | All GPIO_x, 3.3V: High | -32 | – | – | |
| $I_{OH}$ | High-level Output Current with $V_{OH}$ = VDDIO - 0.4V | All GPIO_x, 3.3V: Low | – | – | 2 | mA |
| | | All GPIO_x, 3.3V: Medium | – | – | 20 | |
| | | All GPIO_x, 3.3V: High | – | – | 32 | |
| $I_{IL}$ | Low-level Input Current | LCDPCK, ISC_MCK, GPIO, QSPI_SCK | -1 | – | 1 | µA |
| $I_{IH}$ | High-level Input Current | LCDPCK, ISC_MCK, GPIO, QSPI_SCK | -1 | – | 1 | µA |
| $I_{IL}$ | Low-level Input Current | GPIO_AD | -1 | – | 1 | µA |
| $I_{IH}$ | High-level Input Current | GPIO_AD | -1 | – | 1 | µA |
| $R_{PULLUP}$ | Pull-up Resistor | GPIO_CLK, GPIO_IO, GPIO: 1.8V | 80 | 143 | 310 | kΩ |
| | | GPIO_CLK, GPIO_IO, GPIO: 3.3V | 40 | 66 | 130 | |
| $R_{PULLDOWN}$ | Pull-down Resistor | GPIO_CLK, GPIO_IO, GPIO: 1.8V | 80 | 161 | 430 | kΩ |
| | | GPIO_CLK, GPIO_IO, GPIO: 3.3V | 40 | 77 | 160 | |

.....continued

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $R_{PULLUP}$ | Pull-up Resistor | GPIO_AD: 1.8V | 280 | 380 | 480 | kΩ |
| | | GPIO_AD: 3.3V | 280 | 380 | 480 | |
| $R_{PULLDOWN}$ | Pull-down Resistor | GPIO_AD: 1.8V | 280 | 380 | 480 | kΩ |
| | | GPIO_AD: 3.3V | 280 | 380 | 480 | |
| $R_{SERIAL}$ | Serial Resistor | GPIO | – | 30 | – | Ω |
| | | GPIO_IO | – | 13 | – | Ω |
| | | GPIO_CLK, GPIO_AD | – | 0 | – | Ω |

**Note:**

1. $V_{DDIO}$ voltage must be equal to $V_{DDIN}$ voltage.
2. Current injection may lead to performance degradation or functional failures.

**Table 66-3. I/O Switching Frequency (MHz)**

| GPIO Type | Drive | VDD | | | Unit |
|---|---|---|---|---|---|
| | $C_{Load}$ = 30pF | 1.8V | 2.5V | 3.3V | |
| GPIO_IO | Low | 8 | 12 | 15 | MHz |
| | Medium | 60 | 80 | 90 | |
| | High | 80 | 110 | 110 | |
| GPIO_CLK | Low | 10 | 15 | 18 | |
| | Medium | 90 | 100 | 120 | |
| | High | – | – | – | |
| GPIO_AD | Low | 10 | 15 | 18 | |
| | Medium | 90 | 100 | 120 | |
| | High | – | – | – | |
| GPIO | Low | 7 | 10 | 12 | |
| | Medium | 40 | 50 | 60 | |
| | High | 50 | 60 | 70 | |

**Table 66-4. QSPI I/O Switching Frequency**

| GPIO Type | Description | Name | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|---|
| GPIO_QSPI | Maximum output frequency | $f_{max}$ | Load = 30 pF | – | 133 | MHz |
| | Output duty cycle | – | Load = 30 pF | 45 | 55 | % |

## 66.3 Power Consumption

This section provides information about the current consumption on different power supply rails of the device. It gives current consumption in:

- Active mode when running a CoreMark and in predefined use cases
- Low-power modes: Backup mode, Idle mode and Ultra Low-power mode

- By peripheral with representative activity

## 66.4 Active Mode

Active mode is the normal running mode with the Arm core clock running off a PLL. The power management controller is used to adapt the frequency and to disable the peripheral clocks.

### 66.4.1 Active Mode Power Consumption Versus Modes

The power consumption values are measured under the following operating conditions:

- Parts are from typical process
- $V_{DDIOPx}$ = 3.3V
- $V_{DDSDMMC0}$ and $V_{DDSDMMC1}$ = 1.8V to 3.3V (high frequency)
- $V_{DDCORE}$ = 1.2V +/-2%
- $V_{DDBU}$ = 1.6V to 3.6V
- $T_A$ = as specified in tables "Typical Peripheral Power Consumption by Peripheral in Active Mode" and "Power Consumption in Active Mode: AMP2"
- There is no consumption on the device I/Os.
- All peripheral clocks are disabled.

**Figure 66-1. Measurement Schematics**

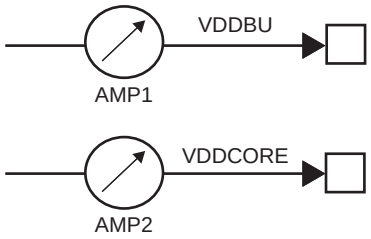



**Table 66-5. Typical Peripheral Power Consumption by Peripheral in Active Mode**

Measurements made at $T_A$ = 25°C and with peripheral clock enabled.

| Peripheral | Clock | Consumption on VDDCORE | Conditions | Consumption (typ) | Unit |
|---|---|---|---|---|---|
| GMAC | MCK/2 | – | See Note 1. | 12 *MCK + 1840*DR (Data rate in Mbits/s) | µA |
| XDMAC0 | MCK | – | See Note 2. | 17.6 * MCK + 4.92 * DR (MCK in MHz, Data rate in Mbytes/s) | |
| XDMAC1 | MCK | – | | | |
| ICM | MCK/2 | 3.52 | – | – | µA/MHz |
| AES | MCK | 8.79 | | | |
| AESB | MCK | 7.49 | | | |
| TDES | MCK/2 | 0.85 | | | |
| SHA | MCK | 3.88 | | | |
| MPDDRC | MCK | 45.21 | See Note 3. | 67 * MCK + 3.11 * DR + 1609 (MCK in MHz, Data rate in Mbytes/s) | µA |

.....continued

| Peripheral | Clock | Consumption on VDDCORE | Conditions | Consumption (typ) | Unit |
|---|---|---|---|---|---|
| HSMC | MCK/2 | 20.12 | – | – | µA/MHz |
| PIOA | MCK/2 | 11.39 | | | |
| FLEXCOM0 | MCK/2 | - | | | |
| FLEX0_USART | | 5.21 | | | |
| FLEX0_SPI | | 7.39 | | | |
| FLEX0_TWI | | 3.88 | | | |
| FLEXCOM1 | MCK/2 | See FLEXCOM0 | | | |
| FLEXCOM2 | MCK/2 | See FLEXCOM0 | | | |
| FLEXCOM3 | MCK/2 | See FLEXCOM0 | | | |
| FLEXCOM4 | MCK/2 | See FLEXCOM0 | | | |
| UART0 | MCK/2 | 1.09 | | | |
| UART1 | MCK/2 | 0.97 | | | |
| UART2 | MCK/2 | 1.21 | | | |
| UART3 | MCK/2 | 0.85 | | | |
| UART4 | MCK/2 | 0.85 | | | |
| TWIHS0 | MCK/2 | 3.27 | | | |
| TWIHS1 | MCK/2 | 3.39 | | | |
| SDMMC0 | MCK | 8.61 | | | |
| SDMMC1 | MCK | 8.61 | | | |
| SPI0 | MCK/2 | 4.61 | | | |
| SPI1 | MCK/2 | 4.48 | | | |
| TC0 | MCK/2 | 3.03 | | | |
| TC1 | MCK/2 | 4 | | | |
| PWM | MCK/2 | 7.03 | | | |
| ADC | MCK/2 | 2.3 | | | |
| UHPHS | MCK/2 | – | See Note 4. | 12 *MCK + 490*DR (MCK in MHz, Data rate in Mbytes/s) | µA |
| UDPHS | MCK/2 | | See Note 5. | 10 *MCK + 206*DR (MCK in MHz, Data rate in Mbytes/s) | |
| SSC0 | MCK/2 | 1.58 | – | – | µA/MHz |
| SSC1 | MCK/2 | 1.58 | | | |
| LCDC | MCK | – | See Note 6. | 9.8 *MCK + 32 *Pix_CLK + 15.7*DR_Baselayer +30.1*DR_Overlayer (MCK & Pixelclock in MHz, Data rate in Mbytes/s); | µA |
| ISC | MCK | | See Note 7. | 10.9*MCK + 22.4*ISPCK + 12.38*DR_sensor (MCK & ISPCK in MHz, Data rate in Mbytes/s) | |
| TRNG | MCK/2 | 760 | – | – | |

.....continued

| Peripheral | Clock | Consumption on VDDCORE | Conditions | Consumption (typ) | Unit |
|---|---|---|---|---|---|
| PDMIC | MCK/2 | 8.24 | – | – | µA/MHz |
| QSPI0 | MCK | 1.94 | | | |
| QSPI1 | MCK | 1.94 | | | |
| I2SC0 | MCK/2 | 0.61 | | | |
| I2SC1 | MCK/2 | 0.61 | | | |
| CAN0 | MCK/2 | 9.7 | | | |
| CAN1 | MCK/2 | 7.27 | | | |

**Note:**

1. In Linux OS, use the ‘iperf’ command to perform bidirectional data transfers. Measure GMAC consumption at different transfer speeds.
2. XDMAC is initialized and one channel performs a memory-to-memory transfer. During test, the data rate is adjusted by changing the DMA setting and the burst size.
3. DDR3 devices are initialized (fully functional). XDMAC performs a memory-to-memory transfer inside the DDR area. Total consumption of MPDDRC and XDMAC is measured. MPDDRC consumption is calculated by discounting XDMAC consumption.
4. In Linux OS, measure UHPHS consumption at different transfer speeds.
5. In Linux OS, build a mass storage using UDPHS. Measure UDPHS consumption at different transfer speeds.
6. The LCD timing engine and each display layer are switched on in sequence. The static image (using random data) is displayed under various resolutions. The 24-bpp RGB888 color space is set for all layers. Auxiliary functions such as rotation, scaling, color space conversion, color look-up table, and chroma upsampling are disabled.
7. ISC performs image sensor preview.

In order to maximize performance, each Peripheral Clock has been timed to H32MX clock frequency. The peripheral frequency can be reduced with the help of a divider in PMC_PCR.

**Table 66-6. Power Consumption in Active Mode: AMP2**

| Conditions $T_A$ = 25°C | | Consumption | |
|---|---|---|---|
| | | Dhrystone (mA) | CoreMark (mA) |
| PLL clock is 1000 MHz, Arm Core clock is 500 MHz, MCK is 166 MHz.<br>– Caches L1 and L2 enabled<br>– Code running off of internal SRAM<br>– Code speed optimization<br>– Run Dhrystone / CoreMark benchmark<br>– Peripheral clock disabled | MRL C<br>MRL B | 114.4 | 108.8 |
| | MRL A | 237.2 | 233.7 |

## 66.5 Low-power Modes

Low-power modes provide a way to balance device power consumption against wake-up time. The modes are described below, in the order of lowest to highest power consumption.

### 66.5.1 Backup Mode

Backup mode achieves the lowest power consumption in the system with limited functionality.

In this mode, only the backup area is powered, maintaining the RTC, the back-up registers, the back-up SRAM and the security module running. This mode is entered by shutting down all the power rails except the VDDBU (refer to the section Shutdown Controller (SHDWC)). To exit Backup mode, the SHDN pin (connected to the enable of the external Power Management IC) must be driven high by an internal event (RTC) or by one of the external events listed below:

- WKUP0 to WKUP9 pins (level transition, configurable debouncing)
- Character received on a serial com receiver (RXLP)
- Analog comparison

Backup mode functionality has been extended with the possibility to keep the DDR memory in Self-refresh state.

### 66.5.2 Backup Mode with DDR in Self-Refresh

Backup mode with DDR in Self-refresh is used to keep the DDR contents when the system is powered off. This mode is achieved by maintaining the backup area and the VDDIODDR powered. To enter Backup Self-refresh mode, follow the sequence below:

- Software saves all the context information to resume (application-dependent).
- Put the DDR in Self-refresh mode and wait until the Self-refresh status is OK (refer to the section Multiport DDR-SDRAM Controller (MPDDRC)).
- Set SFRBU_DDRBUMCR.BUMEN (see SFRBU DDR BU Mode Control Register).
- Enter Backup mode as described above.

To exit this mode, follow the sequence below:

- Once the system is restarted, the software checks the state of SFRBU_DDRBUMCR.BUMEN.
- Restore all the context information.
- Check that the Self-refresh status is OK (refer to the section Multiport DDR-SDRAM Controller (MPDDRC)).
- Disable SFRBU_DDRBUMCR.BUMEN (see SFRBU DDR BU Mode Control Register).
- The DDR memory exits Self-refresh mode when a memory access in the DDR memory space is performed.

### 66.5.3 Ultra Low-power (ULP) Mode

Ultra Low-power mode achieves the lowest power consumption with the system in Retention mode and able to resume on wake-up events (any interrupt or hardware event). This mode is a combination of the Wait for Interrupt mode of the Arm core and a reduced frequency or shut-off of the system clocks.

To obtain the best results, care must be taken that the I/Os (pull-up/pull-down, etc.), USB transceivers, etc., are set to the appropriate state.

The Ultra Low-power mode features two submodes:

- ULP0 mode
- ULP1 mode

#### 66.5.3.1 ULP0 Mode

ULP0 mode maintains a very low frequency clock to wake up on any interrupt.

The selection of the clock depends on the current consumption target versus wake-up time. The higher the frequency, the higher the power consumption.

The sequence to enter ULP0 mode is detailed below. The code used to enter this mode must be executed out of the internal SRAM.

1. Set the DDR to Self-Refresh mode.
2. Set the interrupts to wake up the system.
3. Disable all peripheral clocks.
4. Set the I/Os to an appropriate state and disable the USB transceivers (refer to the section Special Function Registers (SFR)).
5. Switch the system clock to the selected clock (RC12MHZ, 32 KHz Slow Clock.) according to the power consumption and wake-up time awaited (see table VDDCORE Power Consumption in Ultra Low-power Mode: AMP2).
6. Disable the PLLs and all unused clocks (main oscillator, 12 MHz RC oscillator or 32 KHz oscillator).
7. Enter the Wait for Interrupt mode and disable the PCK clock in the PMC_SCDR.

The wake-up from ULP0 mode is triggered by any enabled interrupt. When resuming, the software reconfigures the system (oscillator, PLL, etc.) in the same state as before WFI.

#### 66.5.3.2 ULP1 Mode

Unlike ULP0 mode, all the clocks are off in ULP1 mode, but the number of wake-up sources is limited to the list below:

- WKUP0 pin (level transition, configurable debouncing)
- WKUP1 Secumod wake-up signal
- WKUP2 pin to WKUP9 pin (shared with PIOBU0 to PIOBU7)
- RTC alarm
- USB Resume from Suspend mode
- SDMMC card detect
- RXLP event
- ACC event
- Any SleepWalking event coming from TWI, FLEXCOMx, SPI, or ADC

The sequence to enter the ULP1 mode is detailed below. The code used to enter this mode must be executed out of the internal SRAM.

1. Set the DDR to Self-Refresh mode.
2. Set the events to enable a system wake-up.
3. Disable all peripheral clocks.
4. Set the I/Os to an appropriate state and disable the USB transceivers (refer to the section Special Function Registers (SFR)).
5. Switch the system clock to the 12 MHz RC oscillator.
6. Disable the PLLs and the main oscillator.
7. Enter the ULP1 mode by either:
   7.1. setting the WAITMODE bit in CKGR_MOR, or
   7.2. setting the LPM bit in PMC_FSMR and executing the processor WaitForEvent (WFE) instruction.
8. After setting the WAITMODE bit or using the WFE instruction, wait for the PMC_SR.MCKRDY bit to be set.

### 66.5.4 Idle Mode

In Idle mode, power consumption of the device versus response time is optimized. In this mode, only the core clock is stopped. The peripheral clocks, including the DDR controller clock, can be enabled. The current consumption in this mode is application-dependent and can be reduced by enabling Dynamic Clock Gating (L2CC_POWCR.DCKGATEN = 1).

This mode is entered via the Wait for Interrupt (WFI) instruction and PCK disabling.

The processor can be awakened from an interrupt. The system resumes where it was before entering WFI mode.

### 66.5.5 Low-power Mode Summary Table

The modes detailed above are the main low-power modes. Each part can be set to on or off separately and wake-up sources can be configured individually. The table below gives a summary of the low-power mode configurations.

**Table 66-7. Low-power Mode Configuration Summary**

| Submode | Low-power Mode | | | | |
|---|---|---|---|---|---|
| | Backup | | Ultra-low-power | | Idle |
| | – | Self-refresh | ULP0 | ULP1 | |
| 64 kHz RC Oscillator, 32 kHz Oscillator, RTC, Backup Memory and Registers, POR | ON | | | | |
| 12 MHz RC Oscillator | OFF | | | | ON |
| VDDCORE Regulator | OFF | | ON | | |
| Core | OFF (not powered) | | Powered (not clocked) | | |
| Memory, Peripherals | OFF (not powered) | | Powered (512 Hz) | Powered (not clocked) | Powered (clocked) |
| Mode Entry | Shutdown Controller, FLEXCOM SleepWalking | DDR in Self-refresh, Shutdown Controller | DDR in Self-refresh, Frequency reduced in PMC, WFI | DDR in Self-refresh, CKGR_MOR.WAIT-MODE=1 | DDR in Self-refresh, WFI |
| Potential Wake-up Sources | WKUP0 pin, any PIOBU configured as WKUP pin, RTC alarm, any level above comparator source or character received | Backup mode sources | Any interrupt | Wake-up pins, WOL | Any interrupt |
| Core at Wake-up | Reset | | Clocked back at 512 Hz | Clocked back at 12 MHz | Clocked back at full speed |
| PIO State While in Low-power Mode | Reset | | Previous state saved | | |
| PIO State at Wake-up | Inputs with pull-ups | Unchanged | | | |
| Consumption(2) | $I_{VDDBU}$= 4.5 µA typ(3) at 25°C/3.0V | $I_{VDDBU}$= 4.5 µA typ(3) at 25°C/3.0V $I_{VDDIODDR}$ = 40 µA | 0.21 mA at 25°C/1.1V 0.27 mA at 25°C/1.2V | 0.17 mA at 25°C/1.1V 0.27 mA at 25°C/1.2V | 28 mA at 25°C/ 1.2V(4) |
| Wake-up Time(1) | Start-up time | Start-up time | 300 ms | 15 µs | 800 ns at 498 MHz |

**Note:**

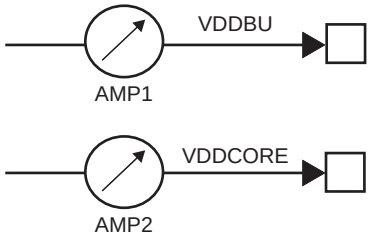
1. When considering wake-up time, the time required to start the PLL is not taken into account. Once started, the device works with the main oscillator. The user has to add the PLL start-up time if it is needed in the system. The wake-up time is defined as the time taken for wake-up until the first instruction is fetched.
2. The external loads on PIOs are not taken into account in the calculation.
3. Total current consumption.
4. Dynamic Clock Gating enabled (L2CC_POWCR.DCKGATEN = 1).

### 66.5.6 Low-power Consumption Versus Modes

The low-power consumption values are measured under the following operating conditions:

- Parts are from typical process
- $V_{DDIOPx}$ = 3.3V
- $V_{DDSDMMC0}$ and $V_{DDSDMMC1}$ = 1.8V to 3.3V (high frequency)
- $V_{DDCORE}$ = 1.2V ±2%
- $V_{DDBU}$ = 1.6V to 3.6V
- $T_A$ = as specified in the tables below
- There is no consumption on the device I/Os.
- All peripheral clocks are disabled.

**Figure 66-2. Measurement Schematics**




In order to maximize performances, each Peripheral Clock has been timed to H32MX clock frequency. The peripheral frequency can be reduced with the help of a divider in PMC_PCR.

**Table 66-8. Typical Power Consumption in Idle Mode: AMP2**

| Conditions | Consumption | | | | |
|---|---|---|---|---|---|
| | $T_A$ 25°C | $T_A$ 70°C | $T_A$ 85°C | $T_A$ 105°C | Unit |
| PLL clock is 1000 MHz, Arm Core clock is 500 MHz, MCK is 166 MHz.<br>– Core clock is stopped<br>– Peripheral clocks, including the DDR Controller clock, can be enabled<br>– Mode is entered via Wait for Interrupt (WFI) instruction and PCK disabling<br>– Measure IDDCORE + IDDBU<br>– Peripheral clock disabled | 28.2 | 29.6 | 30.8 | 33.4 | mA |

**Table 66-9. VDDCORE Power Consumption in Ultra-Low-Power Mode: AMP2**

| Mode | Conditions | Consumption (mA) | | | | Wake-up Time (µs) |
|---|---|---|---|---|---|---|
| | | $T_A$ 25°C | $T_A$ 70°C | $T_A$ 85°C | $T_A$ 105°C | |
| ULP1 Fast Wakeup | Arm Core clock is disabled. MCK is 0. | 0.3 | 1.4 | 2.4 | 4.6 | 15 |
| ULP0 12 MHz | Arm Core clock is disabled. MCK is 12 MHz. | 3.2 | 4.2 | 5.3 | 7.7 | 13 |

.....continued

| Mode | Conditions | Consumption (mA) | | | | Wake-up Time (µs) |
|---|---|---|---|---|---|---|
| | | $T_A$ 25°C | $T_A$ 70°C | $T_A$ 85°C | $T_A$ 105°C | |
| ULP0 750 kHz | Arm Core clock is disabled. MCK is 750 kHz. | 1.6 | 2.6 | 3.7 | 6.1 | 205 |
| ULP0 187 kHz | Arm Core clock is disabled. MCK is 187.5 kHz. | 1.5 | 2.5 | 3.6 | 6.0 | 820 |
| ULP0 32 kHz | Arm Core clock is disabled. MCK is 32 kHz. | 0.3 | 1.5 | 2.6 | 5.0 | 4690 |

**Table 66-10. Typical Power Consumption for Backup Mode**

Measurements made when powered by VDDBU only.

| VDDBU (V) | Conditions | Consumption (µA) | | | | Unit |
|---|---|---|---|---|---|---|
| | | $T_A$ = 25°C | $T_A$ = 70°C | $T_A$ = 85°C | $T_A$ = 105°C | |
| 1.6 | VDDBU Only | 4.2 | 12.1 | 19.3 | 36.8 | µA |
| 1.7 | | 4.2 | 12.1 | 19.3 | 36.9 | |
| 1.8 | | 4.3 | 12.1 | 19.4 | 36.9 | |
| 1.9 | | 4.3 | 12.1 | 19.4 | 36.9 | |
| 2 | | 4.3 | 12.1 | 19.4 | 37 | |
| 2.1 | | 4.3 | 12.2 | 19.4 | 37 | |
| 2.2 | | 4.3 | 12.2 | 19.5 | 37 | |
| 2.3 | | 4.4 | 12.2 | 19.5 | 37 | |
| 2.4 | | 4.4 | 12.2 | 19.5 | 37 | |
| 2.5 | | 4.4 | 12.3 | 19.5 | 37.1 | |
| 2.6 | | 4.4 | 12.3 | 19.6 | 37.1 | |
| 2.7 | | 4.4 | 12.3 | 19.6 | 37.1 | |
| 2.8 | | 4.4 | 12.3 | 19.6 | 37.2 | |
| 2.9 | | 4.5 | 12.4 | 19.6 | 37.2 | |
| 3 | | 4.5 | 12.4 | 19.7 | 37.3 | |
| 3.1 | | 4.5 | 12.5 | 19.8 | 37.7 | |
| 3.2 | | 4.6 | 12.8 | 20.3 | 38.3 | |
| 3.3 | | 4.9 | 13.4 | 20.9 | 38.9 | |
| 3.4 | | 5.5 | 14.1 | 21.6 | 39.7 | |
| 3.5 | | 6.2 | 14.9 | 22.4 | 40.5 | |
| 3.6 | | 7 | 15.7 | 23.3 | 41.4 | |

## 66.6 Clock Characteristics

The following characteristics are applicable to the ambient operating temperature range $T_A$ = -40°C to +105°C for -CN devices, and $T_A$ = -40°C to +85°C for -CU devices, unless otherwise specified.

### 66.6.1 Processor Clock Characteristics

**Table 66-11. Processor Clock Waveform Parameters**

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| 1/($t_{CPPCK}$) | Processor Clock Frequency | VDDCORE[1.1V, 1.32V] | 250(1) | 400 | MHz |
| | | VDDCORE[1.2V, 1.32V] | 250(1) | 500 | |

**Note:**

1. Limitation for DDR2 (125 MHz) usage only. There are no limitations to DDR3, DDR3L, LPDDR1, LPDDR2 and LPDDR3.

### 66.6.2 Master Clock Characteristics

The master clock is the maximum clock at which the system is able to run. It is given by the smallest value of the internal bus clock and EBI clock.

**Table 66-12. Master Clock Waveform Parameters**

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| 1/($t_{CPMCK}$) | Master Clock Frequency | VDDCORE[1.1V, 1.32V] | 125(1) | 133 | MHz |
| | | VDDCORE[1.2V, 1.32V], in DDR2 or LPDDR1 mode, VDDIODDR[1.8V, 1.9V]<br>in LPDDR2 or LPDDR3 mode, VDDIODDR[1.2V, 1.30V]<br>in DDR3 mode, VDDIODDR[1.5V, 1.575V]<br>in DDR3L mode, VDDIODDR[1.35V, 1.45V]<br>Security disabled | 125(1) | 166(2) | |

**Note:**

1. Limitation for DDR2 usage only. There are no limitations to DDR3, DDR3L, LPDDR1, LPDDR2 and LPDDR3.
2. The JEDEC standard specifies a maximum clock frequency of 125 MHz for DDR3 and DDR3L in DLL Off mode. However, check with memory suppliers for higher frequencies.

## 66.7 Oscillator Characteristics

### 66.7.1 Main Oscillator Characteristics

**Table 66-13. 8 to 24 MHz Crystal Oscillator Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $f_{OSC}$ | Operating Frequency | FREQ(2) = 00, 11 | 8 | – | 12 | MHz |
| | | FREQ = 01 | 12 | | 16 | |
| | | FREQ = 10 | 16 | | 24 | |
| – | Duty Cycle | – | 40 | 50 | 60 | % |

.....continued

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $t_{START}$ | Startup Time | FREQ[2] = 00, 11<br>FREQ = 01<br>FREQ = 10 | – | – | 18.5<br>10.5<br>6 | ms |
| $I_{DDON}$ | Current Consumption (on VDDIO) | @ 12 MHz<br>@ 24 MHz | – | 1.2<br>1.7 | 3.5<br>4 | mA |
| $I_{DD_STDBY}$ | Standby Current | – | – | 0.02 | 0.1 | µA |
| $C_{PARA}$ | Internal Parasitic Capacitance[1] | From XIN to XOUT | 1.4 | 1.6 | 1.8 | pF |
| $P_{ON}$ | Drive level | FREQ = 00<br>FREQ = 01, 11<br>FREQ = 10 | – | – | 150<br>300<br>400 | µW |

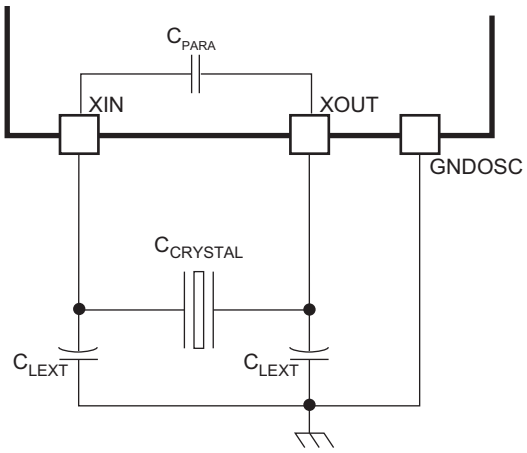
**Note:**

1. The values of external capacitors can be determined by using the following formula:
   $C_{LEXT} = (2 \times C_{CRYSTAL}) - C_{BOARD} - (C_{PARA} \times 2)$

   where:

   - $C_{LEXT}$ is the external capacitor value which must be soldered from XIN to GND and XOUT to GND
   - $C_{CRYSTAL}$ is crystal targeted load
   - $C_{BOARD}$ is the external board parasitic capacitance (from XIN to GND or XOUT to GND)
   - $C_{PARA}$ is the internal parasitic capacitance
2. The SFR_UTMICKTRIM.FREQ field defines the input frequency for the UTMI and the main oscillator. It is important to select the correct FREQ value because this has a direct influence on USB frequency.

**Figure 66-3. Main Oscillator Schematics**



#### 66.7.1.1 Recommended Crystal Characteristics

**Table 66-14. Recommended Crystal Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| ESR | Equivalent Series Resistance | FREQ = 00, 11 | – | – | 100 | Ω |
| | | FREQ = 10, 01 | – | – | 80 | |
| $C_M$ | Motional Capacitance | FREQ = 00 | 5 | – | 9 | fF |
| | | FREQ = 01, 10, 11 | 1.3 | – | 3.2 | |
| $C_S$ | Shunt Capacitance | FREQ = 00, 01, 10 | – | – | 3 | pF |
| | | FREQ = 11 | – | – | 1.3 | |
| $C_{CRYSTAL}$ | Allowed crystal capacitive load | From crystal specification | | | | pF |
| | | FREQ = 00, 01, 11 | 12.5 | – | 18 | |
| | | FREQ = 10 | 8 | – | 12.5 | |

#### 66.7.1.2 XIN Clock Characteristics

These characteristics apply only when the Main Oscillator is in Bypass mode (i.e., when MOSCRCEN = 0 and MOSCXTBY = 1 in CKGR_MOR). Refer to PMC Clock Generator Main Oscillator Register.

**Table 66-15. XIN Clock Electrical Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $1/(t_{CPXIN})$ | XIN Clock Frequency | – | – | – | 50 | MHz |
| $t_{CPXIN}$ | XIN Clock Period | – | 20 | – | – | ns |
| $t_{CHXIN}$ | XIN Clock High Half-period | – | 0.4 x $t_{CPXIN}$ | – | 0.6 x $t_{CPXIN}$ | ns |
| $t_{CLXIN}$ | XIN Clock Low Half-period | – | 0.4 x $t_{CPXIN}$ | – | 0.6 x $t_{CPXIN}$ | ns |
| $C_{IN}$ | XIN Input Capacitance | – | – | – | 25 | pF |
| $R_{IN}$ | XIN Pull-down Resistor | – | – | – | 500 | kΩ |
| $V_{IL_XIN}$ | XIN Low-level Input Voltage | – | – | – | 0.3 x $V_{DDOSC}$ | V |
| $V_{IH_XIN}$ | XIN High-level Input Voltage | – | 0.7 x $V_{DDOSC}$ | – | – | V |

### 66.7.2 12 MHz RC Oscillator Characteristics

**Table 66-16. 12 MHz RC Oscillator Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $f_{OSC}$ | RC Oscillator Frequency | – | 11.4 | – | 12.6 | MHz |
| $t_{START}$ | Startup Time | – | – | – | 15 | µs |
| Duty | Duty Cycle | – | 45 | 50 | 55 | % |
| $I_{DDON}$ | Current Consumption | After startup time | – | 160 | 350 | µA |

### 66.7.3 32.768 kHz Crystal Oscillator Characteristics

**Table 66-17. 32.768 kHz Crystal Oscillator Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $f_{OSC}$ | Operating Frequency | Normal mode with crystal | – | 32.768 | – | kHz |
| $t_{START}$ | Startup Time | Cm> 3fF | – | – | 1200 | ms |

.....continued

| Symbol | Parameter | Conditions | | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| $I_{DDON}$ | Current Consumption | ESR < 50k ohm | $C_{CRYSTAL32}$ = 12.5 pF | – | 440 | 900 | nA |
| | | | $C_{CRYSTAL32}$ = 6 pF | | 600 | 900 | |
| | | ESR < 100k ohm | $C_{CRYSTAL32}$ = 12.5 pF | | 800 | 1200 | |
| | | | $C_{CRYSTAL32}$ = 6 pF | | 700 | 1200 | |
| $C_{PARA32}$ | Internal Parasitic Capacitance | Between XIN32 and XOUT32 | | 1.4 | 1.6 | 1.8 | pF |

**Figure 66-4. 32 kHz Oscillator Schematics**

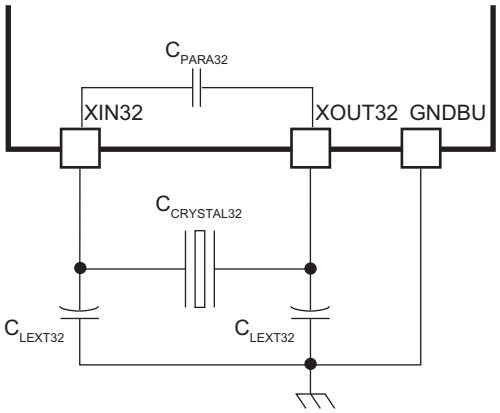



**Table 66-18. Recommended 32.768 kHz Crystal Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| ESR | Equivalent Series Resistor | Crystal at 32.768 kHz | – | – | 100 | kΩ |
| – | Duty Cycle | – | 40 | 50 | 60 | % |
| $C_m$ | Motional Capacitance | Crystal at 32.768 kHz | 3 | – | 8 | fF |
| $C_{SHUNT}$ | Shunt Capacitance | Crystal at 32.768 kHz | 0.6 | – | 2 | pF |
| $C_{CRYSTAL32}$ | Allowed Crystal Capacitance Load(1) | From crystal specification | 6 | – | 12.5 | pF |
| $P_{ON}$ | Drive Level | – | – | – | 0.2 | µW |

**Note:** The value of the external capacitors can be determined by using the following formula:
$C_{LEXT32} = (2 \times C_{CRYSTAL32}) - C_{BOARD} - (C_{PARA32} \times 2)$

where:

- $C_{LEXT32}$ is the external capacitor value which must be soldered from XIN to GND and XOUT to GND
- $C_{CRYSTAL32}$ is crystal targeted load
- $C_{BOARD}$ is the external board parasitic capacitance (from XIN to GND or XOUT to GND)
- $C_{PARA32}$ is the internal parasitic capacitance

### 66.7.4 64 kHz RC Oscillator Characteristics

**Table 66-19. 64 kHz RC Oscillator Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $f_{OSC}$ | RC Oscillator Frequency | – | 40 | – | 90 | kHz |

.....continued

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $t_{START}$ | Startup Time | – | 11 | 17 | 30 | µs |
| $I_{DDON}$ | Current Consumption | After startup time | – | 93 | 140 | nA |

## 66.8 PLL Characteristics

**Table 66-20. PLLA Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $f_{IN}$ | Input Frequency | – | 12 | – | 24 | MHz |
| $f_{OUT}$ | Output Frequency | – | 600 | – | 1200 | MHz |
| $I_{PLL}$ | Current Consumption | Active mode | – | – | 14.5 | mA |
| | | Standby mode | – | – | 2 | µA |
| $t_{START}$ | Startup Time | – | – | – | 60 | µs |

**Table 66-21. UTMI PLL Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $f_{IN}$ | Input Frequency | – | 12 | – | 24 | MHz |
| $f_{OUT}$ | Output Frequency | – | | 480 | | MHz |
| $I_{VDDUTMII}$ | Current Consumption | In Active mode, on VDDUTMII, @480 MHz | – | 6.3 | 7.0 | mA |
| $t_{START}$ | Startup Time | – | – | – | 60 | µs |

**Table 66-22. Audio PLL Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $f_{IN}$ | Input Frequency | – | 12 | – | 24 | MHz |
| $f_{AUDIOCORECLK}$ | AUDIOCORECLK frequency range | – | 620 | – | 700 | MHz |
| $f_{AUDIOPINCLK}$ | AUDIOPINCLK frequency range | – | 8 | 12.288 | 48 | MHz |
| $f_{AUDIOPLLCLK}$ | AUDIOPLLCLK frequency range | – | – | – | 150 | MHz |
| $I_{PLL}$ | Current Consumption | On VDDAUDIOPLL | 6 | – | 20 | mA |
| $t_{START}$ | Startup Time | From OFF to stable AUDIOCORECLK frequency | – | – | 100 | µs |
| $t_{SET}$ | Settling Time(1) | When changing FRACR or NR in PMC_AUDIO_PLL0 or PMC_AUDIO_PLL1 | – | – | 100 | µs |

**Note:** Loop filter is set as recommended in fields BIAS_FILTER and DCO_FILTER of PMC_AUDIO_PLL0.

## 66.9 USB HS Characteristics

### 66.9.1 Electrical Characteristics

The device conforms to all voltage, power, and timing characteristics and specifications set forth in the USB 2.0 Specification. Refer to the USB 2.0 Specification for more information.

### 66.9.2 Dynamic Power Consumption

**Table 66-23. USB Transceiver Dynamic Power Consumption**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $I_{BIAS}$ | Bias Generator Current Consumption | – | – | 0.7 | 0.8 | mA |
| $I_{VDDUTMII}$ | HS Transceiver Current Consumption | HS transmission | – | 47 | 60 | mA |
| | HS Transceiver Current Consumption | HS reception | – | 18 | 27 | mA |
| | LS / FS Transceiver Current Consumption | FS transmission 0m cable(1) | – | 4 | 6 | mA |
| | LS / FS Transceiver Current Consumption | FS transmission 5m cable(1) | – | 26 | 30 | mA |
| | LS / FS Transceiver Current Consumption | FS reception(1) | – | 3 | 4.5 | mA |
| $I_{VDDUTMIC}$ | Core | – | – | 5.5 | 9 | mA |

**Note:** Including 1 mA due to pull-up/pull-down current consumption.

## 66.10 PTC Characteristics

**Table 66-24. PTC Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $C_C$ | Compensation Capacitance | Programmable max code 0x3FFF | 30 | – | – | pF |
| $E_{RS}$ | Error on Serial Filtering Resistance | 20, 50, 100 kOhm | -20 | – | 20 | % |
| $I_{PTCT}$ | PTC Current Consumption | – | – | – | 500 | µA |

## 66.11 ADC Characteristics

The following characteristics are applicable to the ambient operating temperature range TA = -40°C to +105°C for -CN devices, and -40°C to +85°C for -CU devices, unless otherwise specified.

ADVREF is the positive reference of the ADC.

### 66.11.1 ADC Power Supply

#### 66.11.1.1 Power Supply Characteristics

**Table 66-25. Power Supply Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $I_{VDDIN}$ | Analog Current Consumption | Sleep mode(1) | – | 2 | 4 | µA |
| | | Fast Wake-up mode | | 0.4 | 0.6 | mA |
| | | Normal mode, single sampling | | 2.2 | 3.0 | mA |

.....continued

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $I_{VDDCORE}$ | Digital Current Consumption | Sleep mode(1) | – | 1 | 2 | µA |
| | | Normal mode | | 80 | 100 | µA |

**Note:** In Sleep mode, the ADC core, the Sample and Hold and the internal reference operational amplifier are off.

#### 66.11.1.2 ADC Bias Current

The field IBCTL in ADC_ACR controls the ADC biasing current.

The table below gives the IBCTL settings according to the ADC sample rate value.

| IBCTL | Max Data Rate (Ksps) |
|---|---|
| 00 | 250 |
| 01 | 500 |
| 10 | 1000 |
| 11 | 1000 |

### 66.11.2 External Reference Voltage

$V_{ADVREF}$ is an external reference voltage applied on the pin ADVREF. The quality of the reference voltage $V_{ADVREF}$ is critical to the performance of the ADC. A DC variation of the reference voltage $V_{ADVREF}$ is converted to a gain error by the ADC. The noise generated by $V_{ADVREF}$ is converted by the ADC to count noise.

**Table 66-26. ADVREF Electrical Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{ADVREF}$ | Voltage Range | Full operational | 2 | – | VDDANA | V |
| | RMS Noise | Bandwidth 10 kHz to 1 MHz | – | – | 100 | µV |
| $R_{ADVREF}$ | Input DC Impedance | ADC reference resistance bridge(1) | 6 | 8 | 10 | kΩ |
| $I_{ADVREF}$ | Current | $V_{ADVREF}$ = 3.3V | – | – | 460 | µA |

**Note:**

1. When the ADC is off, the ADVREF impedance has a minimum of 1 MΩ.

### 66.11.3 ADC Timings

**Table 66-27. ADC Timing Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $f_{ADC_Clock}$ | Clock Frequency | – | 0.2 | – | 20 | MHz |
| $f_S$ | Sampling Frequency(1) | – | – | – | 1 | MHz |
| $t_{START}$ | ADC Start-up Time | Sleep mode to Normal mode | – | – | 4 | µs |
| | | Fast Wake-up mode to Normal mode | | | 2 | |

**Note:** $t_{ADC_Clock} = 1/f_{ADC_Clock}$ ADC conversion time = 21 $t_{ADC_Clock}$. The Tracking time of the ADC has a minimal value of $t_{TRACKTIM}$ = 15 $t_{ADC_Clock}$.

### 66.11.4 ADC Transfer Function

The DATA code in ADC_CDR is up to 12-bit positive integer or two's complement (signed integer).

#### 66.11.4.1 Differential Mode (12-bit mode)

A differential input voltage $V_{IN} = V_{INP} - V_{INN}$ can be applied between two selected differential pins, e.g. ADC0_AD0 and ADC0_AD1. The ideal code $C_i$ is calculated by using the following formula and rounding the result to the nearest positive integer.

$$C_i = \frac{2047}{V_{\mathrm{ADVREF}}} \times V_{\mathrm{IN}}$$

For the other resolution defined by RES, the code $C_i$ is extended to the corresponding resolution.

The table below is a computation example for the above formula, where $V_{ADVREF}$ = 3V.

**Table 66-28. Input Voltage Values in Differential Mode, Non-signed Output**

| Signed $C_i$ | $V_{IN}$ |
|---|---|
| -2048 | -3 |
| 0 | 0 |
| 2047 | 3 |

#### 66.11.4.2 Single-ended Mode (12-bit mode)

A single input voltage $V_{IN}$ can be applied to selected pins, e.g., ADC0_AD0 or ADC0_AD1. The ideal code $C_i$ is calculated using the following formula and rounding the result to the nearest positive integer.

The single-ended ideal code conversion formula is:

$$C_i = \frac{4095}{V_{\mathrm{ADVREF}}} \times V_{\mathrm{IN}}$$

For the other resolution defined by RES, the code $C_i$ is extended to the corresponding resolution.

The table below is a computation example for the above formula, where $V_{ADVREF}$ = 3V:

**Table 66-29. Input Voltage Values in Single-ended Mode**

| Non-signed $C_i$ | $V_{IN}$ |
|---|---|
| 0 | 0 |
| 2047 | 1.5 |
| 4095 | 3 |

#### 66.11.4.3 Example of LSB Computation

The LSB is relative to the analog scale $V_{ADVREF}$.

The term LSB expresses the quantization step in volts, also used for one ADC code variation.

- Single-ended (SE) (ex: $V_{ADVREF}$ = 3.0V)
  - Gain = 1, LSB = (3.0V / 4096) = 732 µV
- Differential (DIFF) (ex: $V_{ADVREF}$ = 3.0V)
  - Gain = 0.5, LSB = (6.0V / 4096) = 1465 µV

The data include the ADC performances, as the PGA and ADC core cannot be separated. The temperature and voltage dependencies are given as separate parameters.

#### 66.11.4.4 Gain and Offset Errors

For:

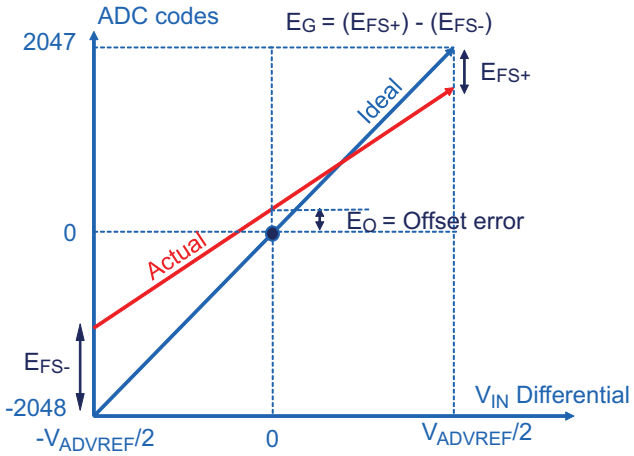
- a given gain error: $E_G$ (%)
- a given ideal code ($C_i$)
- a given offset error: $E_O$ (LSB of 12 bits)
  in 12-bit mode, the actual code ($C_A$) is calculated using the following formula

$$C_A = \left(1 + \frac{E_G}{100}\right) \times (C_i - 2047) + 2047 + E_O$$

**Differential Mode**

In Differential mode, the offset is defined when the differential input voltage is zero.

**Figure 66-5. Gain and Offset Errors in Differential Mode**




where:

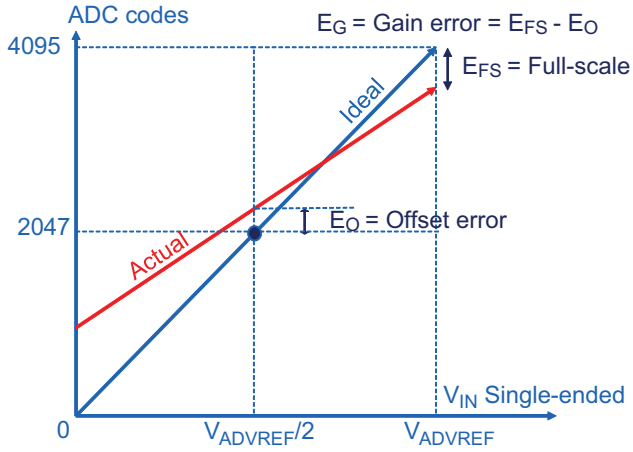
- Full-scale error $E_{FS} = (E_{FS+}) - (E_{FS-})$, unit is LSB code
- Offset error $E_O$ is the offset error measured for $V_{IN}$ = 0V
- Gain error $E_G = 100 \times E_{FS} / 4096$, unit in %

The error values in "ADC Offset and Gain Error, VADVREF = 3.3V" include the sample-and-hold error as well as the PGA gain error.

**Single-ended Mode**

The figure below illustrates the ADC output code relative to an input voltage $V_{IN}$ between 0V (Ground) and $V_{ADVREF}$. The ADC is configured in Single-ended mode by connecting internally the negative differential input to $V_{ADVREF}$ / 2. As the ADC continues to work internally in Differential mode, the offset is measured at $V_{ADVREF}$ / 2. The offset at $V_{INP}$ = 0 can be computed using the transfer function and the corresponding $E_G$ and $E_O$.

**Figure 66-6. Gain and Offset Errors in Single-ended Mode**




where:

- Full-scale error $E_{FS} = (E_{FS+}) - (E_{FS-})$, unit is LSB² code
- Offset error $E_O$ is the offset error measured for $V_{INP} = 0V$
- Gain error $E_G = 100 \times E_{FS} / 2048$, unit in %

The error values in the table "ADC Offset and Gain Error, VADVREF = 3.3V" include the DAC, the sample-and-hold error as well as the PGA gain error.

### 66.11.5 ADC Electrical Characteristics

**Table 66-30. ADC INL and DNL, $V_{ADVREF}$ = 3.3V**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| **Differential Mode** | | | | | | |
| INL | Integral Non-Linearity | – | 1 | – | 1 | LSB |
| DNL | Differential Non-Linearity | – | 1 | – | 1 | LSB |
| **Single-Ended Mode** | | | | | | |
| INL | Integral Non-Linearity | – | 1.5 | – | 1.5 | LSB |
| DNL | Differential Non-Linearity | – | 1 | – | 1 | LSB |

**Table 66-31. ADC Offset and Gain Error, $V_{ADVREF}$ = 3.3V**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| **Differential Mode** | | | | | | |
| $E_O$ | Differential Offset Error | – | -2.0 | – | 2.0 | LSB |
| $E_G$ | Differential Gain Error | – | -0.2 | – | 0.2 | % |
| **Single-Ended Mode** | | | | | | |
| $E_O$ | Single-ended Offset Error | – | -2.0 | – | 2.0 | LSB |
| $E_G$ | Single-ended Gain Error | – | -0.2 | – | 0.2 | % |

**Table 66-32. ADC Analog Input Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{FS}$ | Analog Input Full Scale Range(1) | ADC_COR.DIFFx = 0 | 0 | – | $V_{ADVREF}$ | V |
| | | ADC_COR.DIFFx = 1 | $-V_{ADVREF}$ | – | $V_{ADVREF}$ | |
| $V_{INCM}$ | Common Mode input range(2) | ADC_COR.DIFFx = 1 | 0.4 × $V_{DDANA}$ | – | 0.6 × $V_{DDANA}$ | V |
| $C_{IN}$ | ADC sampling capacitance | – | – | – | 3 | pF |
| $C_{P_ADx}$ | Analog input parasitic capacitance(4) | ADx pin configured as analog input | – | – | 10 | |
| $Z_{IN}$ | Common Mode Input impedance(3) | On ADx pin | 1 / ($f_S$ × $C_{P_ADx}$) | – | – | Ω |

**Note:**

1. $V_{FS} = V_{ADx}$ in Single-ended mode, $V_{FS} = (V_{ADx} - V_{ADx}+1)$ in Differential mode.
2. $V_{INCM} = (V_{ADx} + V_{ADx}+1) / 2$.
3. See the figure "Input Channel Model". When converting one single channel, most of the input parasitic capacitance in not switched, therefore the common mode input impedance reduces to $Z_{IN} = 1 / (f_S \times C_{IN})$.
4. Includes $C_{IN}$.

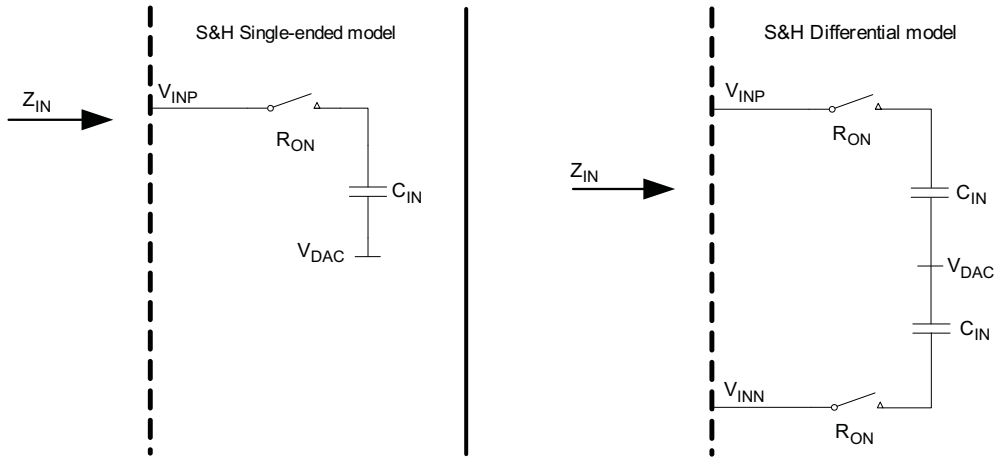
### 66.11.6 Pen Detect Characteristics

The pen detection sensibility is based on a programmable pull-down controlled by the field PENDETSENS of the ADC_ACR register.

| PENDETSENS | Detection Resistor (KOhm) |
|---|---|
| 00 | 200 |
| 01 | 150 |
| 10 | 100 |
| 11 | 50 |

### 66.11.7 ADC Channel Input Impedance

**Figure 66-7. Input Channel Model**




where:

- $Z_{IN}$ is the input impedance in Single-ended or Differential mode

- $C_{IN}$ = 2 pF ±20% depending on the gain value and mode (SE or DIFF); temperature dependency is negligible
- $R_{ON}$ is typical 2 kΩ and 8 kΩ max (worst case process and high temperature)

The following formula is used to calculate input impedance:

$$Z_{\mathrm{IN}} = \frac{1}{f_S \times C_{\mathrm{IN}}}$$

where:

- $f_S$ is the sampling frequency of the ADC channel
- Typ values are used to compute ADC input impedance $Z_{IN}$

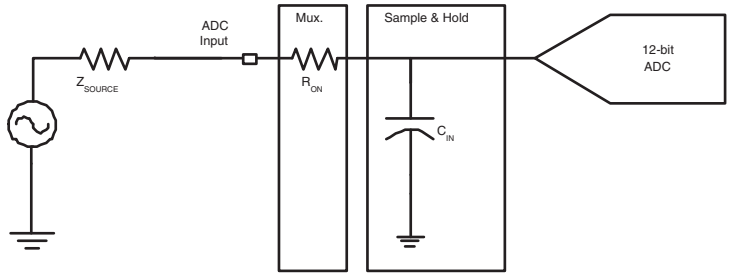
**Table 66-33. $Z_{IN}$ Input Impedance**

| $f_S$ (MHz) | 1 | 0.5 | 0.25 | 0.125 | 0.0625 | 0.03125 | 0.015625 | 0.007813 |
|---|---|---|---|---|---|---|---|---|
| $C_{IN}$ = 2 pF | | | | | | | | |
| $Z_{IN}$ (MΩ) | 0.5 | 1 | 2 | 4 | 8 | 16 | 32 | 64 |

**Track and Hold Time versus Source Output Impedance**

The figure below shows a simplified acquisition path.

**Figure 66-8. Simplified Acquisition Path**




During the tracking phase, the ADC tracks the input signal during the tracking time shown below:

$t_{TRACK} = n \times C_{IN} \times (R_{ON} + Z_{SOURCE}) / 1000$

where

- Tracking time expressed in ns and $Z_{SOURCE}$ expressed in Ω
- n = 8 for 12-bit accuracy
- $R_{ON}$ = 2 kΩ

**Table 66-34. Number of Tau:n**

| Resolution (bits) | 12 |
|---|---|
| RES | 0 |
| n | 8 |

The ADC already includes a tracking time of 15 $t_{ADC\ Clock}$.

## 66.12 Analog Comparator Characteristics

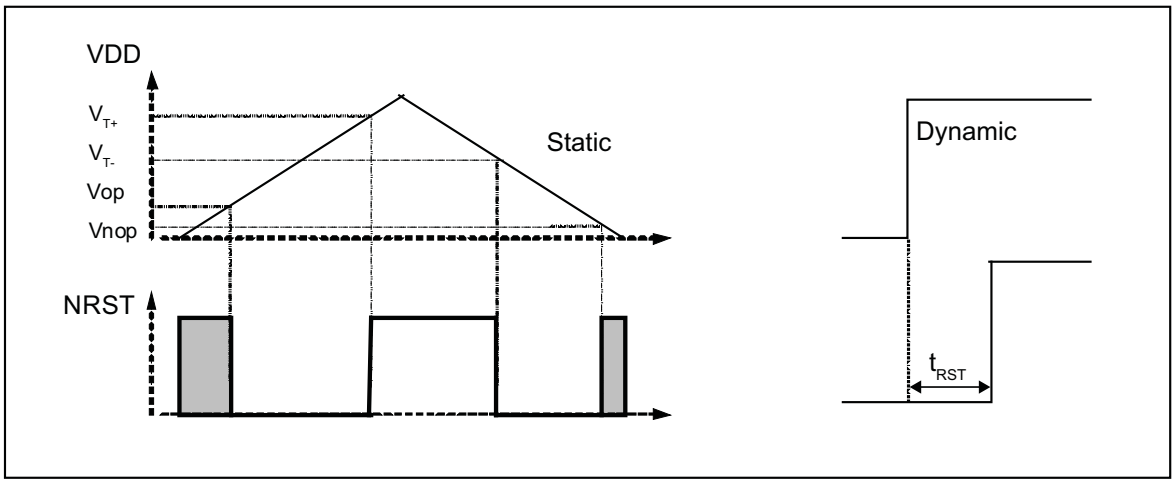
**Table 66-35. Analog Comparator Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{DDBU}$ | Power Supply Voltage Range (VDDBU) | – | 1.65 | 3.3 | 3.6 | V |
| $V_{COMPx}$ | Input Voltage Range | On COMPP or COMPN input | 0 | – | VDDBU | V |
| $V_{hys}$ | Hysteresis | – | 35 | – | 70 | mV |
| TPD | Propagation Delay | 100mV Overdrive | – | – | 350 | µs |
| $f_{in}$ | COMPx Input Signal Frequency | Common mode and differential | – | – | 1 | kHz |
| $I_{VDDBU}$ | Current Consumption (VDDBU) | OFF Mode (ACC_MR.ACEN = 0) | – | – | 50 | nA |
| | | ON Mode (ACC_MR.ACEN = 1) | – | 100 | 200 | |
| $t_{START}$ | Startup Time | – | – | – | 300 | µs |

## 66.13 POR Characteristics

The figure below provides a general presentation of Power-On-Reset (POR) characteristics.

**Figure 66-9. General Presentation of POR Behavior**




When a very slow (versus $t_{RST}$) supply rising slope is applied on the POR VDD pin, the reset time becomes negligible and the reset signal is released when VDD raises higher than $V_{T+}$.

When a very fast (versus $t_{RST}$) supply rising slope is applied on the POR VDD pin, the voltage threshold becomes negligible and the reset signal is released after $t_{RST}$. It is the smallest possible reset time.

**Table 66-36. VDDBU Power-On Reset Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{T+}$ | Threshold Voltage Rising | – | 1.3 | – | 1.5 | V |
| $V_{T-}$ | Threshold Voltage Falling | – | 1.22 | – | 1.4 | V |
| $V_{hys}$ | Hysteresis Voltage | – | 50 | – | 160 | mV |
| $t_{RST}$ | Reset Timeout Period | – | 890 | – | 5100 | µs |

**Table 66-37. VDDCORE Power-On Reset Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{T+}$ | Threshold Voltage Rising | – | 0.927 | – | 1.075 | V |
| $V_{T-}$ | Threshold Voltage Falling | – | 0.848 | – | 1.025 | V |
| $V_{hys}$ | Hysteresis Voltage | – | 38 | – | 109 | mV |
| $t_{RST}$ | Reset Timeout Period | – | 150 | – | 650 | µs |

**Table 66-38. VDDANA Power-On Reset Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{T+}$ | Threshold Voltage Rising | – | 1.3 | – | 1.5 | V |
| $V_{T-}$ | Threshold Voltage Falling | – | 1.22 | – | 1.4 | V |
| $V_{hys}$ | Hysteresis Voltage | – | 50 | – | 160 | mV |
| $t_{RST}$ | Reset Timeout Period | – | 130 | – | 650 | µs |

## 66.14 SMC Timings

### 66.14.1 Timing Conditions

SMC timings are given in max corners.

Timings assuming a capacitance load on data, control and address pads are given in the table below.

**Table 66-39. Capacitance Load**

| Supply | Corner | |
|---|---|---|
| | Max | Min |
| 3.3V | 50 pF | 5 pF |
| 1.8V | 30 pF | 5 pF |

In the tables that follow, $t_{CPMCK}$ is the MCK period.

### 66.14.2 SMC IOSET1 Timing Extraction

#### 66.14.2.1 SMC IOSET1 Read Timings

**Table 66-40. SMC IOSET1 Read Signals - NRD Controlled (READ_MODE = 1)**

| Symbol | Parameter | Min | | Unit |
|---|---|---|---|---|
| | Power supply | 1.8V | 3.3V | |
| NO HOLD SETTINGS (nrd hold = 0) | | | | |
| $SMC_1$ | Data Setup before NRD High | 16.4 | 15 | ns |
| $SMC_2$ | Data Hold after NRD High | 0 | 0 | ns |
| HOLD SETTINGS (nrd hold ≠ 0) | | | | |
| $SMC_3$ | Data Setup before NRD High | 14.4 | 13 | ns |
| $SMC_4$ | Data Hold after NRD High | 0 | 0 | ns |
| HOLD or NO HOLD SETTINGS (nrd hold ≠ 0, nrd hold =0) | | | | |

.....continued

| Symbol | Parameter | Min | | Unit |
|---|---|---|---|---|
| | Power supply | 1.8V | 3.3V | |
| $SMC_5$ | NBS0/A0, NBS1, NBS2/A1, NBS3, A2–A25 Valid before NRD High | (nrd setup + nrd pulse) × $t_{CPMCK}$ | (nrd setup + nrd pulse) × $t_{CPMCK}$ | ns |
| $SMC_6$ | NCS low before NRD High | (nrd setup + nrd pulse - ncs rd setup) × $t_{CPMCK}$ | (nrd setup + nrd pulse - ncs rd setup) × $t_{CPMCK}$ | ns |
| $SMC_7$ | NRD Pulse Width | nrd pulse × $t_{CPMCK}$ | nrd pulse × $t_{CPMCK}$ | ns |

**Table 66-41. SMC IOSET1 Read Signals - NCS Controlled (READ_MODE = 0)**

| Symbol | Parameter | Min | | Unit |
|---|---|---|---|---|
| | Power supply | 1.8V | 3.3V | |
| NO HOLD SETTINGS (ncs rd hold = 0) | | | | |
| $SMC_8$ | Data Setup before NCS High | 17.9 | 15.7 | ns |
| $SMC_9$ | Data Hold after NCS High | 0 | 0 | ns |
| HOLD SETTINGS (ncs rd hold ≠ 0) | | | | |
| $SMC_{10}$ | Data Setup before NCS High | 15.9 | 13.7 | ns |
| $SMC_{11}$ | Data Hold after NCS High | 0 | 0 | ns |
| HOLD or NO HOLD SETTINGS (ncs rd hold ≠ 0, ncs rd hold = 0) | | | | |
| $SMC_{12}$ | NBS0/A0, NBS1, NBS2/A1, NBS3, A2–A25 valid before NCS High | (ncs rd setup + ncs rd pulse) × $t_{CPMCK}$ | (ncs rd setup + ncs rd pulse) × $t_{CPMCK}$ | ns |
| $SMC_{13}$ | NRD low before NCS High | (ncs rd setup + ncs rd pulse - nrd setup) × $t_{CPMCK}$ | (ncs rd setup + ncs rd pulse - nrd setup) × $t_{CPMCK}$ | ns |
| $SMC_{14}$ | NCS Pulse Width | ncs rd pulse length × $t_{CPMCK}$ | ncs rd pulse length × $t_{CPMCK}$ | ns |

### 66.14.2.2 SMC IOSET1 Write Timings

**Table 66-42. SMC IOSET1 Write Signals - NWE Controlled (WRITE_MODE = 1)**

| Symbol | Parameter | Min | | Unit |
|---|---|---|---|---|
| | Power supply | 1.8V | 3.3V | |
| HOLD or NO HOLD SETTINGS (nwe hold ≠ 0, nwe hold = 0) | | | | |
| $SMC_{15}$ | Data Out Valid before NWE High | nwe pulse × $t_{CPMCK}$ | nwe pulse × $t_{CPMCK}$ | ns |
| $SMC_{16}$ | NWE Pulse Width | nwe pulse × $t_{CPMCK}$ | nwe pulse × $t_{CPMCK}$ | ns |
| $SMC_{17}$ | NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25 valid before NWE low | nwe setup × $t_{CPMCK}$ | nwe pulse × $t_{CPMCK}$ | ns |
| $SMC_{18}$ | NCS low before NWE high | (nwe setup - ncs rd setup + nwe pulse) × $t_{CPMCK}$ | (nwe setup - ncs rd setup + nwe pulse) × $t_{CPMCK}$ | ns |
| HOLD SETTINGS (nwe hold ≠ 0) | | | | |
| $SMC_{19}$ | NWE High to Data OUT, NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25 change | nwe hold × $t_{CPMCK}$ | nwe hold × $t_{CPMCK}$ | ns |
| $SMC_{20}$ | NWE High to NCS Inactive(1) | (nwe hold - ncs wr hold) × $t_{CPMCK}$ | (nwe hold - ncs wr hold) × $t_{CPMCK}$ | ns |
| NO HOLD SETTINGS (nwe hold = 0) | | | | |

.....continued

| Symbol | Parameter | Min | | Unit |
|---|---|---|---|---|
| | Power supply | 1.8V | 3.3V | |
| $SMC_{21}$ | NWE High to Data OUT, NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25, NCS change(1) | 2.3 | 1.3 | ns |

**Note:** hold length = total cycle duration - setup duration - pulse duration. “hold length” is for “ncs wr hold length” or “NWE hold length”.

**Table 66-43. SMC IOSET1 Write NCS Controlled (WRITE_MODE = 0)**

| Symbol | Parameter | Min | | Unit |
|---|---|---|---|---|
| | Power supply | 1.8V | 3.3V | |
| $SMC_{22}$ | Data Out Valid before NCS High | ncs wr pulse × $t_{CPMCK}$ | ncs wr pulse × $t_{CPMCK}$ | ns |
| $SMC_{23}$ | NCS Pulse Width | $SMC_{14}$ | $SMC_{14}$ | ns |
| $SMC_{24}$ | NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25 valid before NCS low | ncs wr setup × $t_{CPMCK}$ | ncs wr setup × $t_{CPMCK}$ | ns |
| $SMC_{25}$ | NWE low before NCS high | (ncs wr setup - nwe setup + ncs pulse) × $t_{CPMCK}$ | (ncs wr setup - nwe setup + ncs pulse) × $t_{CPMCK}$ | ns |
| $SMC_{26}$ | NCS High to Data Out, NBS0/A0, NBS1, NBS2/A1, NBS3, A2–A25, change | ncs wr hold × $t_{CPMCK}$ | ncs wr hold × $t_{CPMCK}$ | ns |
| $SMC_{27}$ | NCS High to NWE Inactive | (ncs wr hold - nwe hold) × $t_{CPMCK}$ | (ncs wr hold - nwe hold) × $t_{CPMCK}$ | ns |

### 66.14.3 SMC IOSET2 Timing Extraction

#### 66.14.3.1 SMC IOSET2 Read Timings

**Table 66-44. SMC IOSET2 Read Signals - NRD Controlled (READ_MODE = 1)**

| Symbol | Parameter | Min | | Unit |
|---|---|---|---|---|
| | Power supply | 1.8V | 3.3V | |
| NO HOLD SETTINGS (nrd hold = 0) | | | | |
| $SMC_1$ | Data Setup before NRD High | 16.5 | 15 | ns |
| $SMC_2$ | Data Hold after NRD High | 0 | 0 | ns |
| HOLD SETTINGS (nrd hold ≠ 0) | | | | |
| $SMC_3$ | Data Setup before NRD High | 14 | 12.6 | ns |
| $SMC_4$ | Data Hold after NRD High | 0 | 0 | ns |
| HOLD or NO HOLD SETTINGS (nrd hold ≠ 0, nrd hold =0) | | | | |
| $SMC_5$ | NBS0/A0, NBS1, NBS2/A1, NBS3, A2–A25 Valid before NRD High | (nrd setup + nrd pulse) × $t_{CPMCK}$ | (nrd setup + nrd pulse) × $t_{CPMCK}$ | ns |
| $SMC_6$ | NCS low before NRD High | (nrd setup + nrd pulse - ncs rd setup) × $t_{CPMCK}$ | (nrd setup + nrd pulse - ncs rd setup) × $t_{CPMCK}$ | ns |
| $SMC_7$ | NRD Pulse Width | nrd pulse × $t_{CPMCK}$ | nrd pulse × $t_{CPMCK}$ | ns |

**Table 66-45. SMC IOSET2 Read Signals - NCS Controlled (READ_MODE = 0)**

| Symbol | Parameter | Min | | Unit |
|---|---|---|---|---|
| | Power supply | 1.8V | 3.3V | |
| NO HOLD SETTINGS (ncs rd hold = 0) | | | | |
| $SMC_8$ | Data Setup before NCS High | 18.1 | 15.7 | ns |
| $SMC_9$ | Data Hold after NCS High | 0 | 0 | ns |
| HOLD SETTINGS (ncs rd hold ≠ 0) | | | | |
| $SMC_{10}$ | Data Setup before NCS High | 15.7 | 13.3 | ns |
| $SMC_{11}$ | Data Hold after NCS High | 0 | 0 | ns |
| HOLD or NO HOLD SETTINGS (ncs rd hold ≠ 0, ncs rd hold = 0) | | | | |
| $SMC_{12}$ | NBS0/A0, NBS1, NBS2/A1, NBS3, A2–A25 valid before NCS High | (ncs rd setup + ncs rd pulse) × $t_{CPMCK}$ | (ncs rd setup + ncs rd pulse) × $t_{CPMCK}$ | ns |
| $SMC_{13}$ | NRD low before NCS High | (ncs rd setup + ncs rd pulse - nrd setup) × $t_{CPMCK}$ | (ncs rd setup + ncs rd pulse - nrd setup) × $t_{CPMCK}$ | ns |
| $SMC_{14}$ | NCS Pulse Width | ncs rd pulse length × $t_{CPMCK}$ | ncs rd pulse length × $t_{CPMCK}$ | ns |

### 66.14.3.2 SMC IOSET2 Write Timings

**Table 66-46. SMC IOSET2 Write Signals - NWE Controlled (WRITE_MODE = 1)**

| Symbol | Parameter | Min | | Unit |
|---|---|---|---|---|
| | Power supply | 1.8V | 3.3V | |
| HOLD or NO HOLD SETTINGS (nwe hold ≠ 0, nwe hold = 0) | | | | |
| $SMC_{15}$ | Data Out Valid before NWE High | nwe pulse × $t_{CPMCK}$ | nwe pulse × $t_{CPMCK}$ | ns |
| $SMC_{16}$ | NWE Pulse Width | nwe pulse × $t_{CPMCK}$ | nwe pulse × $t_{CPMCK}$ | ns |
| $SMC_{17}$ | NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25 valid before NWE low | nwe setup × $t_{CPMCK}$ | nwe pulse × $t_{CPMCK}$ | ns |
| $SMC_{18}$ | NCS low before NWE high | (nwe setup - ncs rd setup + nwe pulse) × $t_{CPMCK}$ | (nwe setup - ncs rd setup + nwe pulse) × $t_{CPMCK}$ | ns |
| HOLD SETTINGS (nwe hold ≠ 0) | | | | |
| $SMC_{19}$ | NWE High to Data OUT, NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25 change | nwe hold × $t_{CPMCK}$ | nwe hold × $t_{CPMCK}$ | ns |
| $SMC_{20}$ | NWE High to NCS Inactive(1) | (nwe hold - ncs wr hold) × $t_{CPMCK}$ | (nwe hold - ncs wr hold) × $t_{CPMCK}$ | ns |
| NO HOLD SETTINGS (nwe hold = 0) | | | | |
| $SMC_{21}$ | NWE High to Data OUT, NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25, NCS change(1) | 1.2 | 0.6 | ns |

**Note:** hold length = total cycle duration - setup duration - pulse duration. “hold length” is for “ncs wr hold length” or “NWE hold length”.

**Table 66-47. SMC IOSET2 Write NCS Controlled (WRITE_MODE = 0)**

| Symbol | Parameter | Min | | Unit |
|---|---|---|---|---|
| | Power supply | 1.8V | 3.3V | |
| $SMC_{22}$ | Data Out Valid before NCS High | ncs wr pulse × $t_{CPMCK}$ | ncs wr pulse × $t_{CPMCK}$ | ns |

.....continued

| Symbol | Parameter | Min | | Unit |
|---|---|---|---|---|
| | Power supply | 1.8V | 3.3V | |
| $SMC_{23}$ | NCS Pulse Width | $SMC_{14}$ | $SMC_{14}$ | ns |
| $SMC_{24}$ | NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25 valid before NCS low | ncs wr setup × $t_{CPMCK}$ | ncs wr setup × $t_{CPMCK}$ | ns |
| $SMC_{25}$ | NWE low before NCS high | (ncs wr setup - nwe setup + ncs pulse) × $t_{CPMCK}$ | (ncs wr setup - nwe setup + ncs pulse) × $t_{CPMCK}$ | ns |
| $SMC_{26}$ | NCS High to Data Out, NBS0/A0, NBS1, NBS2/A1, NBS3, A2–A25, change | ncs wr hold × $t_{CPMCK}$ | ncs wr hold × $t_{CPMCK}$ | ns |
| $SMC_{27}$ | NCS High to NWE Inactive | (ncs wr hold - nwe hold) × $t_{CPMCK}$ | (ncs wr hold - nwe hold) × $t_{CPMCK}$ | ns |

**Figure 66-10. SMC Timings - NCS Controlled Read and Write**

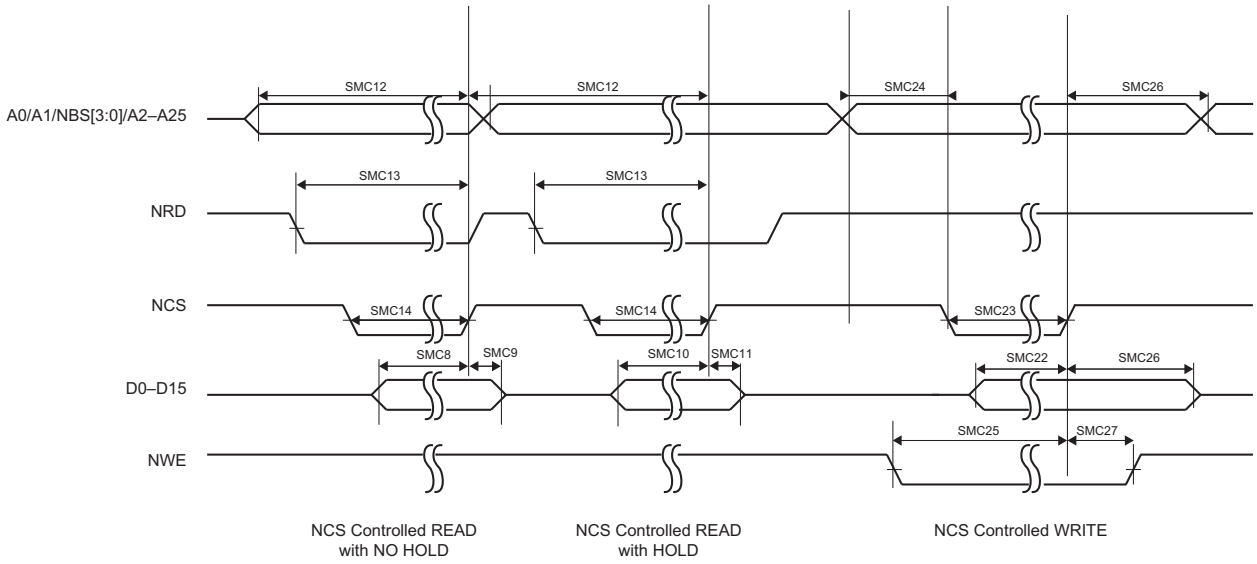
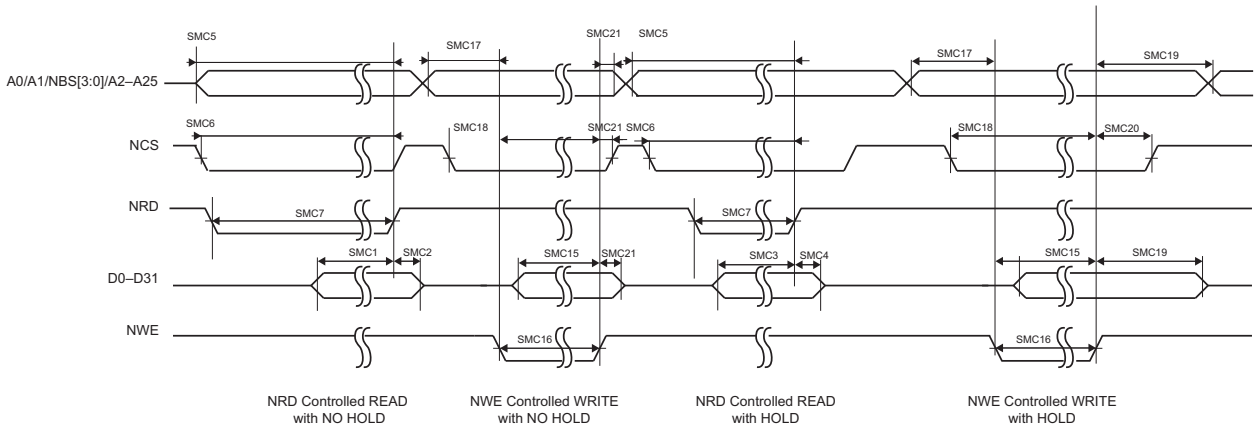



**Figure 66-11. SMC Timings - NRD Controlled Read and NWE Controlled Write**



## 66.15 FLEXCOM Timings

### 66.15.1 FLEXCOM USART in Asynchronous Modes

Refer to the section USART in Asynchronous Modes.

### 66.15.2 FLEXCOM SPI Timings

#### 66.15.2.1 Timing Conditions

Timings assuming a capacitance load on MISO, SPCK and MOSI are given in the table "Capacitance Load for MISO, SPCK and MOSI (SPI0 and SPI1)".

**Table 66-48. Capacitance Load for MISO, SPCK and MOSI (FLEXCOM 0, 1, 2, 3, 4)**

| Supply | Corner | |
|---|---|---|
| | Max | Min |
| 3.3V | 40 pF | 5 pF |
| 1.8V | 20 pF | 5 pF |

#### 66.15.2.2 Timing Extraction

**Figure 66-12. FLEXCOM in SPI Master Modes 1 and 2**

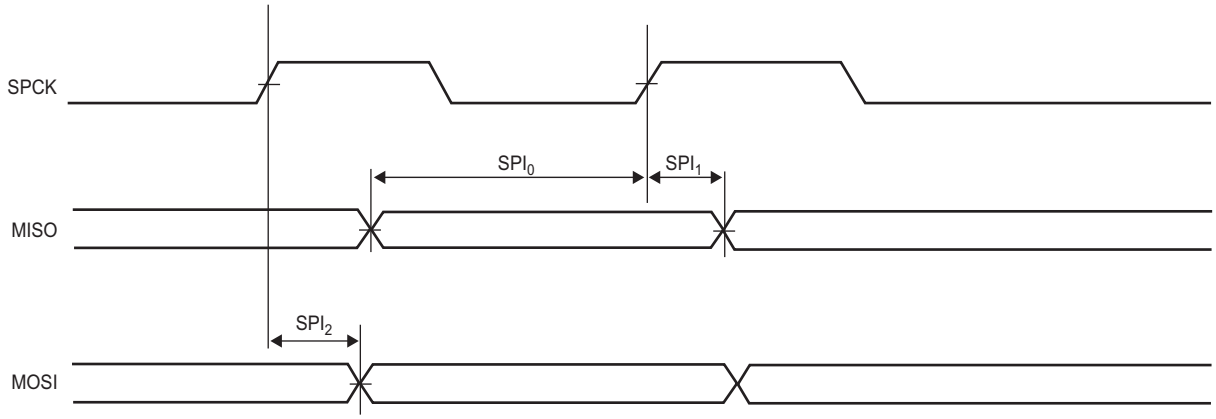



**Figure 66-13. FLEXCOM in SPI Master Modes 0 and 3**

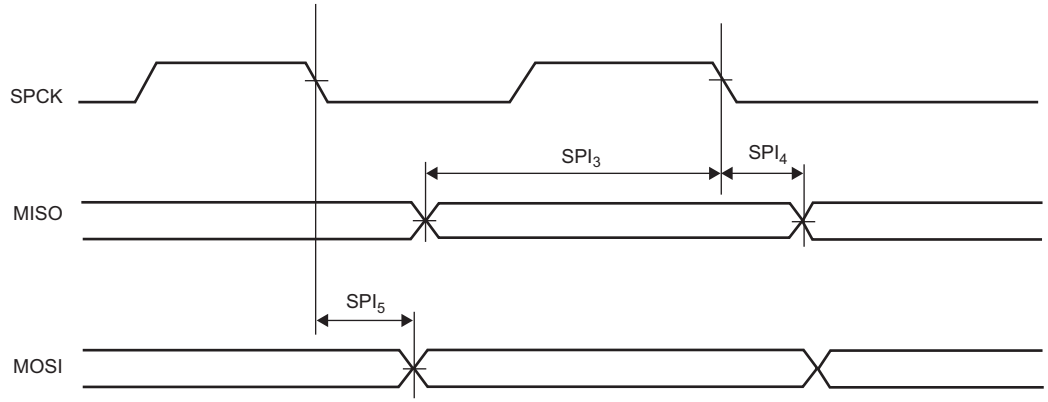


**Figure 66-14. FLEXCOM in SPI Slave Modes 0 and 3**

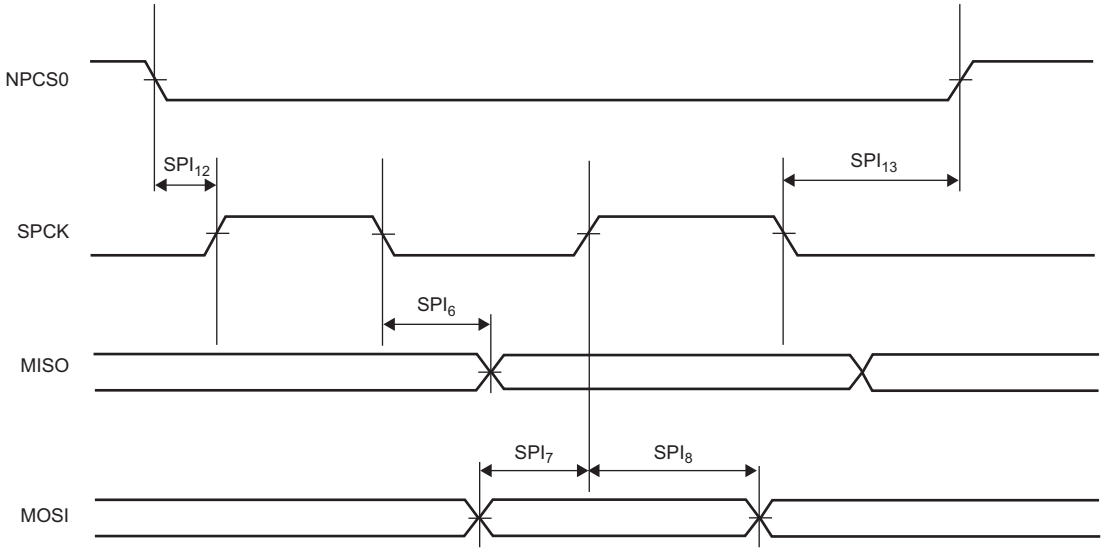



**Figure 66-15. FLEXCOM in SPI Slave Modes 1 and 2**

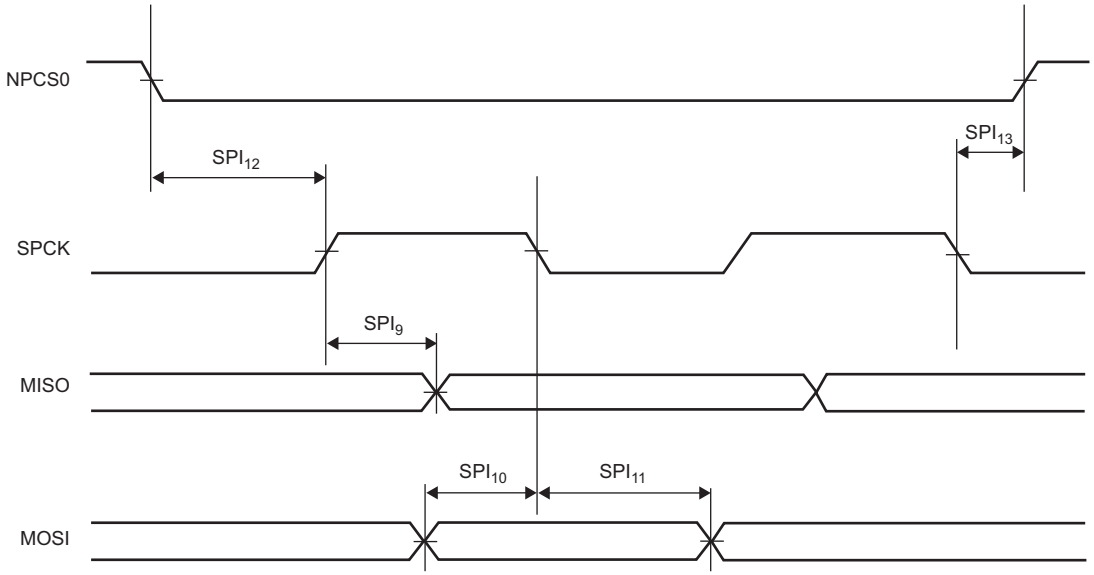



**Figure 66-16. FLEXCOM in SPI Slave Mode - NPCS Timings**

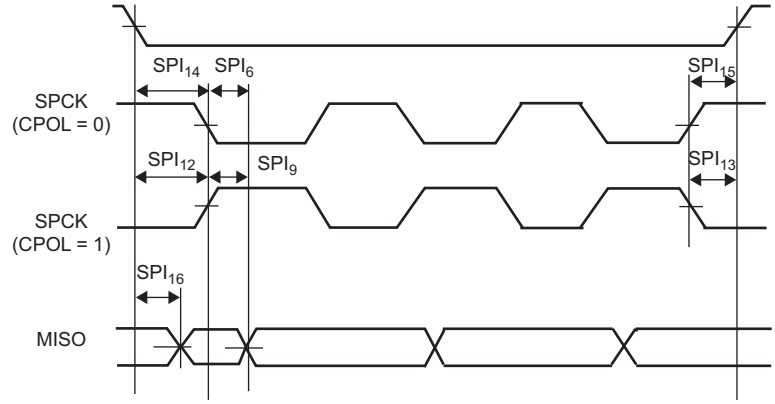


**Table 66-49. FLEXCOM0 in SPI Mode IOSET1 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Master Mode | | | | | |
| $SPI_0$ | MISO Setup time before SPCK rises | 14.3 | – | 12.8 | – | ns |
| $SPI_1$ | MISO Hold time after SPCK rises | 0 | – | 0 | – | ns |
| $SPI_2$ | SPCK rising to MOSI | 0 | 3.9 | 0 | 4.2 | ns |
| $SPI_3$ | MISO Setup time before SPCK falls | 14.7 | – | 13.4 | – | ns |
| $SPI_4$ | MISO Hold time after SPCK falls | 0 | – | 0 | – | ns |
| $SPI_5$ | SPCK falling to MOSI | 0 | 3.9 | 0 | 4.6 | ns |
| | Slave Mode | | | | | |
| $SPI_6$ | SPCK falling to MISO | 10.9 | 13.4 | 9.1 | 11.9 | ns |
| $SPI_7$ | MOSI Setup time before SPCK rises | 5.3 | – | 5 | – | ns |
| $SPI_8$ | MOSI Hold time after SPCK rises | 0.4 | – | 0.3 | – | ns |
| $SPI_9$ | SPCK rising to MISO | 10.7 | 13.1 | 9 | 11.5 | ns |
| $SPI_{10}$ | MOSI Setup time before SPCK falls | 5.3 | – | 5 | – | ns |
| $SPI_{11}$ | MOSI Hold time after SPCK falls | 0.4 | – | 0.3 | – | ns |
| $SPI_{12}$ | NPCS0 setup to SPCK rising | 5.6 | – | 5.4 | – | ns |
| $SPI_{13}$ | NPCS0 hold after SPCK falling | 0.7 | – | 0.6 | – | ns |
| $SPI_{14}$ | NPCS0 setup to SPCK falling | 5.4 | – | 5.2 | – | ns |
| $SPI_{15}$ | NPCS0 hold after SPCK rising | 0.2 | – | 0.1 | – | ns |
| $SPI_{16}$ | NPCS0 falling to MISO valid | 17.5 | – | 16.2 | – | ns |

**Table 66-50. FLEXCOM1 in SPI Mode IOSET1 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Master Mode | | | | | |
| $SPI_0$ | MISO Setup time before SPCK rises | 15.7 | – | 13.8 | – | ns |
| $SPI_1$ | MISO Hold time after SPCK rises | 0 | – | 0 | – | ns |
| $SPI_2$ | SPCK rising to MOSI | 0 | 2.5 | 0 | 2.6 | ns |
| $SPI_3$ | MISO Setup time before SPCK falls | 15.2 | – | 13.9 | – | ns |
| $SPI_4$ | MISO Hold time after SPCK falls | 0 | – | 0 | – | ns |
| $SPI_5$ | SPCK falling to MOSI | 0 | 1.7 | 0 | 2.4 | ns |
| | Slave Mode | | | | | |
| $SPI_6$ | SPCK falling to MISO | 11.4 | 13.7 | 9.4 | 12.3 | ns |
| $SPI_7$ | MOSI Setup time before SPCK rises | 2.4 | – | 2.1 | – | ns |
| $SPI_8$ | MOSI Hold time after SPCK rises | 1 | – | 0.9 | – | ns |
| $SPI_9$ | SPCK rising to MISO | 11 | 13.1 | 9 | 11.6 | ns |

.....continued

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| $SPI_{10}$ | MOSI Setup time before SPCK falls | 2.4 | – | 2.1 | – | ns |
| $SPI_{11}$ | MOSI Hold time after SPCK falls | 1 | – | 0.9 | – | ns |
| $SPI_{12}$ | NPCS0 setup to SPCK rising | 3.9 | – | 3.7 | – | ns |
| $SPI_{13}$ | NPCS0 hold after SPCK falling | 1.1 | – | 1 | – | ns |
| $SPI_{14}$ | NPCS0 setup to SPCK falling | 3.4 | – | 3.3 | – | ns |
| $SPI_{15}$ | NPCS0 hold after SPCK rising | 0.6 | – | 0.4 | – | ns |
| $SPI_{16}$ | NPCS0 falling to MISO valid | 16.8 | – | 15.5 | – | ns |

**Table 66-51. FLEXCOM2 in SPI Mode IOSET1 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Master Mode | | | | | |
| $SPI_0$ | MISO Setup time before SPCK rises | 15.3 | – | 13.4 | – | ns |
| $SPI_1$ | MISO Hold time after SPCK rises | 0 | – | 0 | – | ns |
| $SPI_2$ | SPCK rising to MOSI | 0 | 2.5 | 0 | 3 | ns |
| $SPI_3$ | MISO Setup time before SPCK falls | 15.6 | – | 13.7 | – | ns |
| $SPI_4$ | MISO Hold time after SPCK falls | 0 | – | 0 | – | ns |
| $SPI_5$ | SPCK falling to MOSI | 0 | 2.6 | 0 | 3.1 | ns |
| | Slave Mode | | | | | |
| $SPI_6$ | SPCK falling to MISO | 11.7 | 13.5 | 9.4 | 11.7 | ns |
| $SPI_7$ | MOSI Setup time before SPCK rises | 2 | – | 1.6 | – | ns |
| $SPI_8$ | MOSI Hold time after SPCK rises | 0.5 | – | 0.5 | – | ns |
| $SPI_9$ | SPCK rising to MISO | 11.7 | 13.5 | 9.4 | 11.5 | ns |
| $SPI_{10}$ | MOSI Setup time before SPCK falls | 2 | – | 1.6 | – | ns |
| $SPI_{11}$ | MOSI Hold time after SPCK falls | 0.5 | – | 0.5 | – | ns |
| $SPI_{12}$ | NPCS0 setup to SPCK rising | 4 | – | 3.7 | – | ns |
| $SPI_{13}$ | NPCS0 hold after SPCK falling | 0.6 | – | 0.6 | – | ns |
| $SPI_{14}$ | NPCS0 setup to SPCK falling | 3.9 | – | 3.7 | – | ns |
| $SPI_{15}$ | NPCS0 hold after SPCK rising | 0.4 | – | 0.3 | – | ns |
| $SPI_{16}$ | NPCS0 falling to MISO valid | 16.8 | – | 13.6 | – | ns |

**Table 66-52. FLEXCOM2 in SPI Mode IOSET2 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Master Mode | | | | | |
| $SPI_0$ | MISO Setup time before SPCK rises | 13.3 | – | 11.2 | – | ns |

.....continued

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| $SPI_1$ | MISO Hold time after SPCK rises | 0 | – | 0 | – | ns |
| $SPI_2$ | SPCK rising to MOSI | 0 | 4.4 | 0 | 4.1 | ns |
| $SPI_3$ | MISO Setup time before SPCK falls | 4.3 | – | 12.5 | – | ns |
| $SPI_4$ | MISO Hold time after SPCK falls | 0 | – | 0 | – | ns |
| $SPI_5$ | SPCK falling to MOSI | 0.3 | 4.4 | 0.4 | 4.4 | ns |
| | Slave Mode | | | | | |
| $SPI_6$ | SPCK falling to MISO | 10.1 | 12.2 | 8.2 | 10.4 | ns |
| $SPI_7$ | MOSI Setup time before SPCK rises | 3.3 | – | 3.1 | – | ns |
| $SPI_8$ | MOSI Hold time after SPCK rises | 0.8 | – | 0.7 | – | ns |
| $SPI_9$ | SPCK rising to MISO | 9.8 | 11.8 | 7.9 | 9.8 | ns |
| $SPI_{10}$ | MOSI Setup time before SPCK falls | 3.3 | – | 3.1 | – | ns |
| $SPI_{11}$ | MOSI Hold time after SPCK falls | 0.8 | – | 0.7 | – | ns |
| $SPI_{12}$ | NPCS0 setup to SPCK rising | 5.3 | – | 5.2 | – | ns |
| $SPI_{13}$ | NPCS0 hold after SPCK falling | 0.8 | – | 0.6 | – | ns |
| $SPI_{14}$ | NPCS0 setup to SPCK falling | 5 | – | 4.9 | – | ns |
| $SPI_{15}$ | NPCS0 hold after SPCK rising | 0.2 | – | 0.1 | – | ns |
| $SPI_{16}$ | NPCS0 falling to MISO valid | 15.9 | – | 14.2 | – | ns |

**Table 66-53. FLEXCOM3 in SPI Mode IOSET1 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Master Mode | | | | | |
| $SPI_0$ | MISO Setup time before SPCK rises | 14.6 | – | 12.7 | – | ns |
| $SPI_1$ | MISO Hold time after SPCK rises | 0 | – | 0 | – | ns |
| $SPI_2$ | SPCK rising to MOSI | 0 | 2.6 | 0 | 2.9 | ns |
| $SPI_3$ | MISO Setup time before SPCK falls | 14.2 | – | 13 | – | ns |
| $SPI_4$ | MISO Hold time after SPCK falls | 0 | – | 0 | – | ns |
| $SPI_5$ | SPCK falling to MOSI | 0 | 1.8 | 0 | 2.9 | ns |
| | Slave Mode | | | | | |
| $SPI_6$ | SPCK falling to MISO | 11.6 | 14.1 | 9.5 | 12.7 | ns |
| $SPI_7$ | MOSI Setup time before SPCK rises | 3.3 | – | 3.2 | – | ns |
| $SPI_8$ | MOSI Hold time after SPCK rises | 0.9 | – | 0.7 | – | ns |
| $SPI_9$ | SPCK rising to MISO | 11.2 | 13.6 | 9.1 | 12.2 | ns |
| $SPI_{10}$ | MOSI Setup time before SPCK falls | 3.3 | – | 3.2 | – | ns |
| $SPI_{11}$ | MOSI Hold time after SPCK falls | 0.9 | – | 0.7 | – | ns |
| $SPI_{12}$ | NPCS0 setup to SPCK rising | 2.8 | – | 2.6 | – | ns |

.....continued

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| $SPI_{13}$ | NPCS0 hold after SPCK falling | 1.5 | – | 1.3 | – | ns |
| $SPI_{14}$ | NPCS0 setup to SPCK falling | 2.3 | – | 2.2 | – | ns |
| $SPI_{15}$ | NPCS0 hold after SPCK rising | 1 | – | 0.7 | – | ns |
| $SPI_{16}$ | NPCS0 falling to MISO valid | 15.4 | – | 14.1 | – | ns |

**Table 66-54. FLEXCOM3 in SPI Mode IOSET2 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Master Mode | | | | | |
| $SPI_0$ | MISO Setup time before SPCK rises | 14.1 | – | 12.6 | – | ns |
| $SPI_1$ | MISO Hold time after SPCK rises | 0 | – | 0 | – | ns |
| $SPI_2$ | SPCK rising to MOSI | 0 | 3.8 | 0 | 4.1 | ns |
| $SPI_3$ | MISO Setup time before SPCK falls | 14.9 | – | 13.7 | – | ns |
| $SPI_4$ | MISO Hold time after SPCK falls | 0 | – | 0 | – | ns |
| $SPI_5$ | SPCK falling to MOSI | 0 | 3.9 | 0 | 4.5 | ns |
| | Slave Mode | | | | | |
| $SPI_6$ | SPCK falling to MISO | 10.6 | 13.1 | 8.6 | 11.8 | ns |
| $SPI_7$ | MOSI Setup time before SPCK rises | 3.7 | – | 3.5 | – | ns |
| $SPI_8$ | MOSI Hold time after SPCK rises | 0.6 | – | 0.5 | – | ns |
| $SPI_9$ | SPCK rising to MISO | 10.2 | 12.6 | 8.2 | 11.1 | ns |
| $SPI_{10}$ | MOSI Setup time before SPCK falls | 3.7 | – | 3.5 | – | ns |
| $SPI_{11}$ | MOSI Hold time after SPCK falls | 0.6 | – | 0.5 | – | ns |
| $SPI_{12}$ | NPCS0 setup to SPCK rising | 4.5 | – | 4.3 | – | ns |
| $SPI_{13}$ | NPCS0 hold after SPCK falling | 1 | – | 0.9 | – | ns |
| $SPI_{14}$ | NPCS0 setup to SPCK falling | 4 | – | 3.9 | – | ns |
| $SPI_{15}$ | NPCS0 hold after SPCK rising | 0.4 | – | 0.3 | – | ns |
| $SPI_{16}$ | NPCS0 falling to MISO valid | 16.9 | – | 15.6 | – | ns |

**Table 66-55. FLEXCOM3 in SPI Mode IOSET3 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Master Mode | | | | | |
| $SPI_0$ | MISO Setup time before SPCK rises | 14.2 | – | 12.7 | – | ns |
| $SPI_1$ | MISO Hold time after SPCK rises | 0 | – | 0 | – | ns |
| $SPI_2$ | SPCK rising to MOSI | 0 | 3.4 | 0 | 3.7 | ns |
| $SPI_3$ | MISO Setup time before SPCK falls | 15.1 | – | 13.8 | – | ns |

.....continued

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| $SPI_4$ | MISO Hold time after SPCK falls | 0 | – | 0 | – | ns |
| $SPI_5$ | SPCK falling to MOSI | 0 | 3.5 | 0 | 4.2 | ns |
| | Slave Mode | | | | | |
| $SPI_6$ | SPCK falling to MISO | 11.4 | 14.1 | 9.4 | 12.8 | ns |
| $SPI_7$ | MOSI Setup time before SPCK rises | 5.4 | – | 5.1 | – | ns |
| $SPI_8$ | MOSI Hold time after SPCK rises | 0.4 | – | 0.3 | – | ns |
| $SPI_9$ | SPCK rising to MISO | 11 | 13.6 | 9 | 12.2 | ns |
| $SPI_{10}$ | MOSI Setup time before SPCK falls | 5.4 | – | 5.1 | – | ns |
| $SPI_{11}$ | MOSI Hold time after SPCK falls | 0.4 | – | 0.3 | – | ns |
| $SPI_{12}$ | NPCS0 setup to SPCK rising | 4.3 | – | 4.2 | – | ns |
| $SPI_{13}$ | NPCS0 hold after SPCK falling | 1.1 | – | 0.9 | – | ns |
| $SPI_{14}$ | NPCS0 setup to SPCK falling | 3.8 | – | 3.7 | – | ns |
| $SPI_{15}$ | NPCS0 hold after SPCK rising | 0.5 | – | 0.3 | – | ns |
| $SPI_{16}$ | NPCS0 falling to MISO valid | 17.5 | – | 16.2 | – | ns |

**Table 66-56. FLEXCOM4 in SPI Mode IOSET1 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Master Mode | | | | | |
| $SPI_0$ | MISO Setup time before SPCK rises | 14.7 | – | 13.1 | – | ns |
| $SPI_1$ | MISO Hold time after SPCK rises | 0 | – | 0 | – | ns |
| $SPI_2$ | SPCK rising to MOSI | 0 | 2.7 | 0 | 3.2 | ns |
| $SPI_3$ | MISO Setup time before SPCK falls | 15.2 | – | 13.8 | – | ns |
| $SPI_4$ | MISO Hold time after SPCK falls | 0 | – | 0 | – | ns |
| $SPI_5$ | SPCK falling to MOSI | 0 | 2.9 | 0 | 3.6 | ns |
| | Slave Mode | | | | | |
| $SPI_6$ | SPCK falling to MISO | 10.9 | 13.2 | 8.9 | 11.9 | ns |
| $SPI_7$ | MOSI Setup time before SPCK rises | 3.3 | – | 3.2 | – | ns |
| $SPI_8$ | MOSI Hold time after SPCK rises | 0.7 | – | 0.6 | – | ns |
| $SPI_9$ | SPCK rising to MISO | 10.5 | 12.7 | 8.5 | 11.3 | ns |
| $SPI_{10}$ | MOSI Setup time before SPCK falls | 3.3 | – | 3.2 | – | ns |
| $SPI_{11}$ | MOSI Hold time after SPCK falls | 0.7 | – | 0.6 | – | ns |
| $SPI_{12}$ | NPCS0 setup to SPCK rising | 5.7 | – | 5.5 | – | ns |
| $SPI_{13}$ | NPCS0 hold after SPCK falling | 0.6 | – | 0.5 | – | ns |
| $SPI_{14}$ | NPCS0 setup to SPCK falling | 5.2 | – | 5 | – | ns |
| $SPI_{15}$ | NPCS0 hold after SPCK rising | 0 | – | 0 | – | ns |

.....continued

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| $SPI_{16}$ | NPCS0 falling to MISO valid | 17.8 | – | 16.5 | – | ns |

**Table 66-57. FLEXCOM4 in SPI Mode IOSET2 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Master Mode | | | | | |
| $SPI_0$ | MISO Setup time before SPCK rises | 14.7 | – | 11.5 | – | ns |
| $SPI_1$ | MISO Hold time after SPCK rises | 0 | – | 0 | – | ns |
| $SPI_2$ | SPCK rising to MOSI | 0 | 2.7 | 0 | 4.7 | ns |
| $SPI_3$ | MISO Setup time before SPCK falls | 15.2 | – | 12.8 | – | ns |
| $SPI_4$ | MISO Hold time after SPCK falls | 0 | – | 0 | – | ns |
| $SPI_5$ | SPCK falling to MOSI | 0 | 2.9 | 0.8 | 5.1 | ns |
| | Slave Mode | | | | | |
| $SPI_6$ | SPCK falling to MISO | 10.9 | 13.2 | 7.8 | 10.2 | ns |
| $SPI_7$ | MOSI Setup time before SPCK rises | 3.3 | – | 2.2 | – | ns |
| $SPI_8$ | MOSI Hold time after SPCK rises | 0.7 | – | 0.8 | – | ns |
| $SPI_9$ | SPCK rising to MISO | 10.5 | 12.7 | 7.7 | 9.9 | ns |
| $SPI_{10}$ | MOSI Setup time before SPCK falls | 3.3 | – | 2.2 | – | ns |
| $SPI_{11}$ | MOSI Hold time after SPCK falls | 0.7 | – | 0.8 | – | ns |
| $SPI_{12}$ | NPCS0 setup to SPCK rising | 5.6 | – | 3.3 | – | ns |
| $SPI_{13}$ | NPCS0 hold after SPCK falling | 0.6 | – | 1.1 | – | ns |
| $SPI_{14}$ | NPCS0 setup to SPCK falling | 5.2 | – | 3.1 | – | ns |
| $SPI_{15}$ | NPCS0 hold after SPCK rising | 0 | – | 0.8 | – | ns |
| $SPI_{16}$ | NPCS0 falling to MISO valid | 17.8 | – | 12.9 | – | ns |

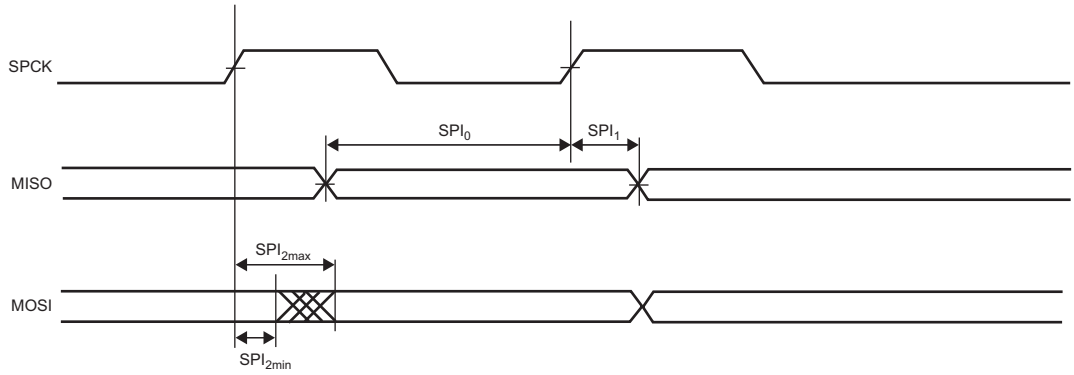
**Table 66-58. FLEXCOM4 in SPI Mode IOSET3 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Master Mode | | | | | |
| $SPI_0$ | MISO Setup time before SPCK rises | 14.2 | – | 12.2 | – | ns |
| $SPI_1$ | MISO Hold time after SPCK rises | 0 | – | 0 | – | ns |
| $SPI_2$ | SPCK rising to MOSI | 0 | 3.6 | 0 | 3.6 | ns |
| $SPI_3$ | MISO Setup time before SPCK falls | 15.1 | – | 13.3 | – | ns |
| $SPI_4$ | MISO Hold time after SPCK falls | 0 | – | 0 | – | ns |
| $SPI_5$ | SPCK falling to MOSI | 0.1 | 3.7 | 0.1 | 4 | ns |
| | Slave Mode | | | | | |

.....continued

| Symbol | Parameter | Power Supply | 1.8V | | 3.3V | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| $SPI_6$ | SPCK falling to MISO | | 9.9 | 12.4 | 8 | 10.5 | ns |
| $SPI_7$ | MOSI Setup time before SPCK rises | | 3.9 | – | 3.7 | – | ns |
| $SPI_8$ | MOSI Hold time after SPCK rises | | 0.8 | – | 0.7 | – | ns |
| $SPI_9$ | SPCK rising to MISO | | 9.5 | 11.9 | 7.7 | 9.9 | ns |
| $SPI_{10}$ | MOSI Setup time before SPCK falls | | 3.9 | – | 3.7 | – | ns |
| $SPI_{11}$ | MOSI Hold time after SPCK falls | | 0.8 | – | 0.7 | – | ns |
| $SPI_{12}$ | NPCS0 setup to SPCK rising | | 4.8 | – | 4.6 | – | ns |
| $SPI_{13}$ | NPCS0 hold after SPCK falling | | 1 | – | 0.9 | – | ns |
| $SPI_{14}$ | NPCS0 setup to SPCK falling | | 4.4 | – | 4.3 | – | ns |
| $SPI_{15}$ | NPCS0 hold after SPCK rising | | 0.5 | – | 0.3 | – | ns |
| $SPI_{16}$ | NPCS0 falling to MISO valid | | 16 | – | 14.2 | – | ns |

**Figure 66-17. Minimum and Maximum Access Time for SPI Output Signal**




### 66.15.3 FLEXCOM TWI Timings

Refer to the section TWI Timings.

## 66.16 USART in Asynchronous Modes

In Asynchronous modes, the maximum baud rate that can be achieved is MCK2 / 8, if the bit USART_MR.OVER=1.

Example: if MCK2 = 83 MHz, the baud rate is 10.375 Mbit/s.

## 66.17 SPI Timings

### 66.17.1 Maximum SPI Frequency

The following formulas give maximum SPI frequency in Master Read and Write modes and in Slave Read and Write modes.

- Master Write Mode
  The SPI sends data to a slave device only, e.g. an LCD. The limit is given by $SPI_2$ (or $SPI_5$) timing.

- Master Read Mode

$$f_{SPCK}\max = \frac{1}{SPI_0(\text{or } SPI_3) + t_{VALID}}$$

$t_{VALID}$ is the slave time response to output data after deleting an SPCK edge.

The $f_{SPCK}$ max is given between the maximum frequency given by the above formula and the pad I/O limitation.

- Slave Read Mode
In Slave mode, SPCK is the input clock for the SPI. The max SPCK frequency is given by setup and hold timings $SPI_7$/$SPI_8$ (or $SPI_{10}$/$SPI_{11}$). Since this gives a frequency well above the pad limit, the limit in Slave Read mode is given by the SPCK pad.
- Slave Write Mode

$$f_{SPCK}\max = \frac{1}{SPI_6(\text{or } SPI_9) + t_{SETUP}}$$

$t_{SETUP}$ is the setup time from the master before sampling data (6 ns).

The $f_{SPCK}$ max is given between the maximum frequency given by the above formula and the pad I/O limitation.

### 66.17.2 Timing Conditions

Timings assuming a capacitance load on MISO, SPCK and MOSI are given in the table below.

**Table 66-59. Capacitance Load for MISO, SPCK and MOSI (SPI0 and SPI1)**

| Supply | Corner | |
|---|---|---|
| | Max | Min |
| 3.3V | 40 pF | 5 pF |
| 1.8V | 20 pF | 5 pF |

### 66.17.3 Timing Extraction

In the first two figures below, “SPI Master Modes 1 and 2” and “SPI Master Modes 0 and 3”, the MOSI line shifting edge is represented with a hold time = 0. However, it is important to note that for this device, the MISO line is sampled prior to the MOSI line shifting edge. As shown in “MISO Capture in Master Mode”, the device sampling point extends the propagation delay ($t_p$) for slave and routing delays to more than half the SPI clock period, whereas the common sampling point allows only less than half the SPI clock period.

As an example, an SPI Slave working in Mode 0 is safely driven if the SPI Master is configured in Mode 0.

**Figure 66-18. MISO Capture in Master Mode**

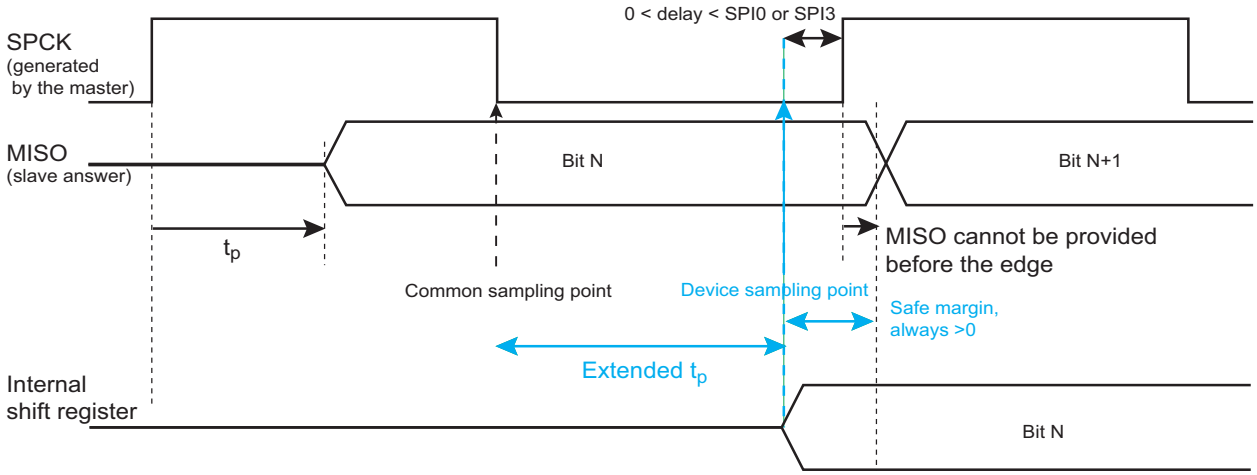



**Figure 66-19. SPI Master Modes 1 and 2**

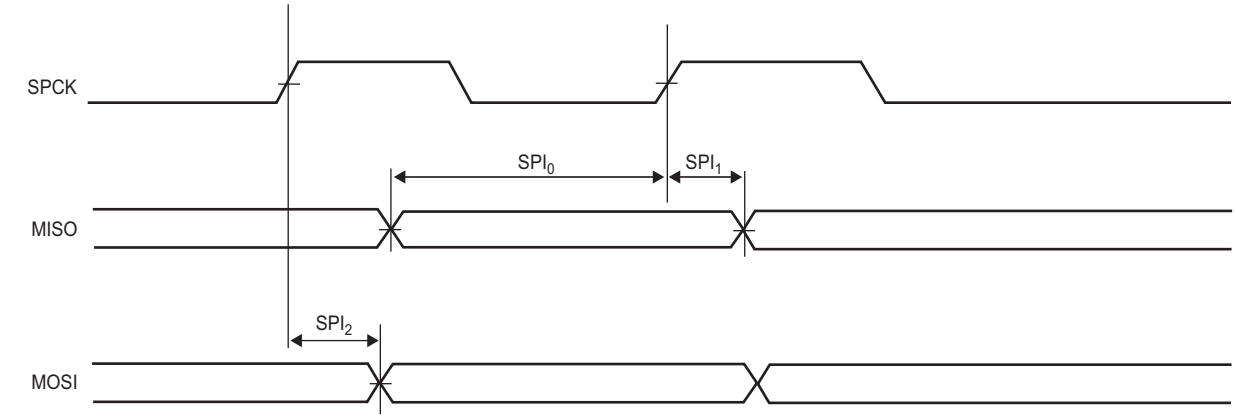



**Figure 66-20. SPI Master Modes 0 and 3**

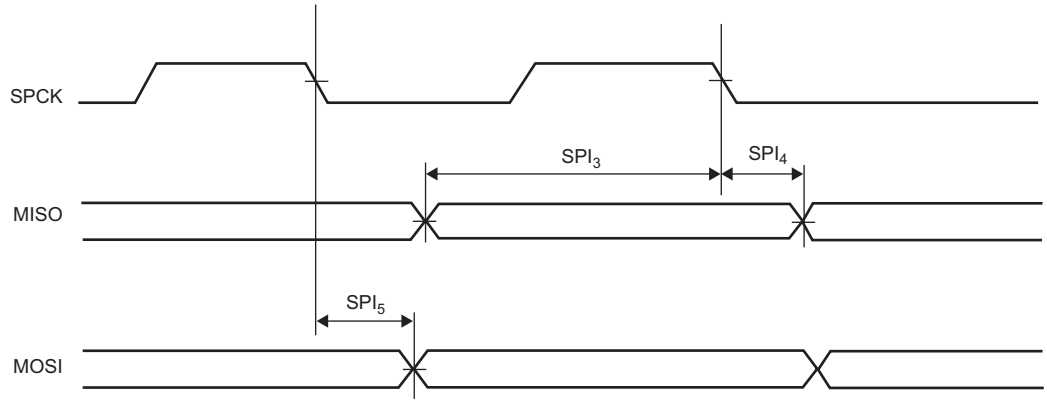


**Figure 66-21. SPI Slave Modes 0 and 3**

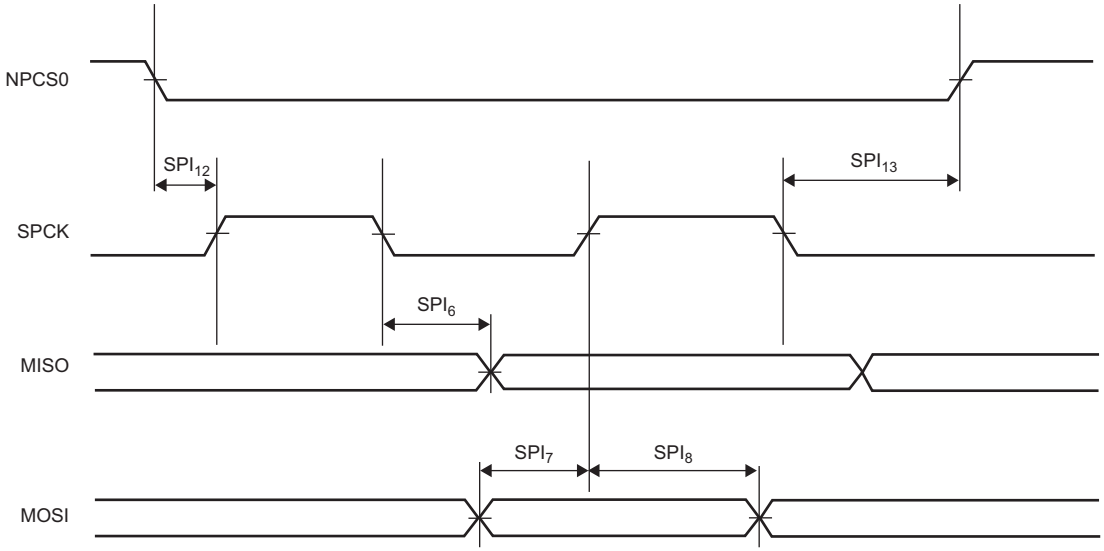



**Figure 66-22. SPI Slave Modes 1 and 2**

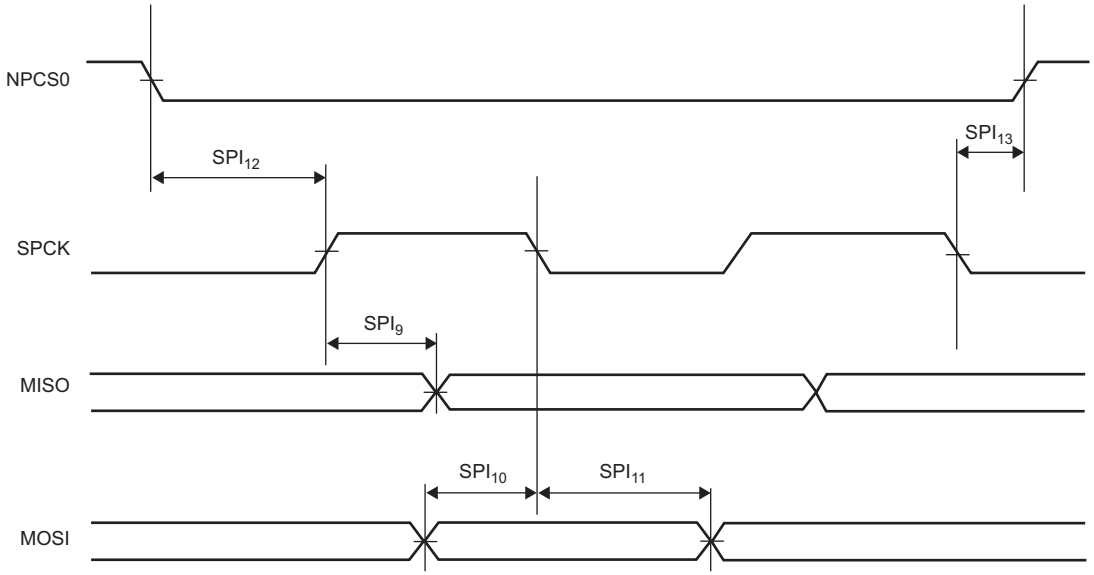



**Figure 66-23. SPI Slave Mode - NPCS Timings**

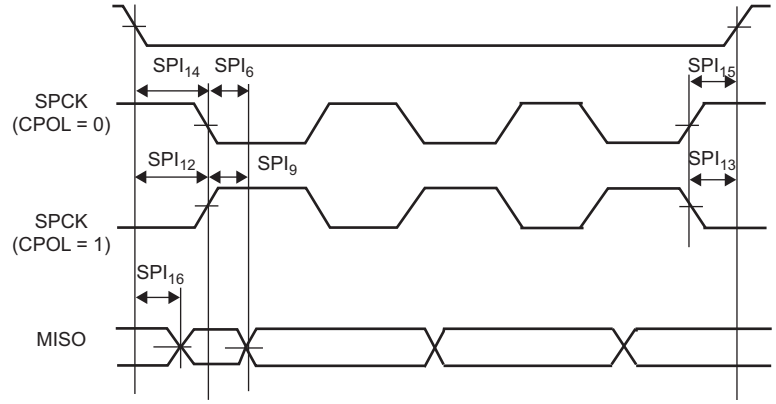


**Table 66-60. SPI0 IOSET1 Timings**

| Symbol | Power Supply | 1.8V | | 3.3V | | Unit |
|---|---|---|---|---|---|---|
| | Parameter | Min | Max | Min | Max | |
| Master Mode | | | | | | |
| $SPI_0$ | MISO Setup time before SPCK rises | 14.3 | – | 12.4 | – | ns |
| $SPI_1$ | MISO Hold time after SPCK rises | 0 | – | 0 | – | ns |
| $SPI_2$ | SPCK rising to MOSI | 0 | 1.9 | 0 | 2.4 | ns |
| $SPI_3$ | MISO Setup time before SPCK falls | 13.8 | – | 12.6 | – | ns |
| $SPI_4$ | MISO Hold time after SPCK falls | 0 | – | 0 | – | ns |
| $SPI_5$ | SPCK falling to MOSI | 0 | 1.2 | 0 | 2.3 | ns |
| Slave Mode | | | | | | |
| $SPI_6$ | SPCK falling to MISO | 10.5 | 12.6 | 8.4 | 10.9 | ns |
| $SPI_7$ | MOSI Setup time before SPCK rises | 1.5 | – | 1.4 | – | ns |
| $SPI_8$ | MOSI Hold time after SPCK rises | 1.7 | – | 1.5 | – | ns |
| $SPI_9$ | SPCK rising to MISO | 10 | 12 | 8 | 10.2 | ns |
| $SPI_{10}$ | MOSI Setup time before SPCK falls | 1.5 | – | 1.4 | – | ns |
| $SPI_{11}$ | MOSI Hold time after SPCK falls | 1.7 | – | 1.5 | – | ns |
| $SPI_{12}$ | NPCS0 setup to SPCK rising | 4.4 | – | 4.3 | – | ns |
| $SPI_{13}$ | NPCS0 hold after SPCK falling | 1.5 | – | 1.3 | – | ns |
| $SPI_{14}$ | NPCS0 setup to SPCK falling | 3.9 | – | 3.9 | – | ns |
| $SPI_{15}$ | NPCS0 hold after SPCK rising | 0.8 | – | 0.5 | – | ns |
| $SPI_{16}$ | NPCS0 falling to MISO valid | 13.3 | – | 11.7 | – | ns |

**Table 66-61. SPI0 IOSET2 Timings**

| Symbol | Power Supply | 1.8V | | 3.3V | | Unit |
|---|---|---|---|---|---|---|
| | Parameter | Min | Max | Min | Max | |
| Master Mode | | | | | | |
| $SPI_0$ | MISO Setup time before SPCK rises | 14.5 | – | 13 | – | ns |
| $SPI_1$ | MISO Hold time after SPCK rises | 0 | – | 0 | – | ns |
| $SPI_2$ | SPCK rising to MOSI | 0 | 2.5 | 0 | 3 | ns |
| $SPI_3$ | MISO Setup time before SPCK falls | 14.9 | – | 13.6 | – | ns |
| $SPI_4$ | MISO Hold time after SPCK falls | 0 | – | 0 | – | ns |
| $SPI_5$ | SPCK falling to MOSI | 0 | 2.7 | 0 | 3.4 | ns |
| Slave Mode | | | | | | |
| $SPI_6$ | SPCK falling to MISO | 10.3 | 12. | 8.5 | 11.2 | ns |
| $SPI_7$ | MOSI Setup time before SPCK rises | 2.3 | – | 2.2 | – | ns |
| $SPI_8$ | MOSI Hold time after SPCK rises | 1 | – | 0.9 | – | ns |
| $SPI_9$ | SPCK rising to MISO | 9.9 | 12 | 8.1 | 10.5 | ns |

.....continued

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| $SPI_{10}$ | MOSI Setup time before SPCK falls | 2.3 | – | 2.2 | – | ns |
| $SPI_{11}$ | MOSI Hold time after SPCK falls | 1 | – | 0.9 | – | ns |
| $SPI_{12}$ | NPCS0 setup to SPCK rising | 6.1 | – | 5.9 | – | ns |
| $SPI_{13}$ | NPCS0 hold after SPCK falling | 0.8 | – | 0.7 | – | ns |
| $SPI_{14}$ | NPCS0 setup to SPCK falling | 5.6 | – | 5.6 | – | ns |
| $SPI_{15}$ | NPCS0 hold after SPCK rising | 0.2 | – | 0.1 | – | ns |
| $SPI_{16}$ | NPCS0 falling to MISO valid | 15.1 | – | 13.8 | – | ns |

**Table 66-62. SPI1 IOSET1 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Master Mode | | | | | |
| $SPI_0$ | MISO Setup time before SPCK rises | 14.6 | – | 13.1 | – | ns |
| $SPI_1$ | MISO Hold time after SPCK rises | 0 | – | 0 | – | ns |
| $SPI_2$ | SPCK rising to MOSI | 0 | 0.8 | 0 | 1.2 | ns |
| $SPI_3$ | MISO Setup time before SPCK falls | 15 | – | 13.7 | – | ns |
| $SPI_4$ | MISO Hold time after SPCK falls | 0 | – | 0 | – | ns |
| $SPI_5$ | SPCK falling to MOSI | 0 | 0.9 | 0 | 1.6 | ns |
| | Slave Mode | | | | | |
| $SPI_6$ | SPCK falling to MISO | 10.3 | 12.4 | 8.3 | 11.2 | ns |
| $SPI_7$ | MOSI Setup time before SPCK rises | 3.5 | – | 3.4 | – | ns |
| $SPI_8$ | MOSI Hold time after SPCK rises | 0.8 | – | 0.7 | – | ns |
| $SPI_9$ | SPCK rising to MISO | 9.8 | 11.8 | 7.8 | 10.4 | ns |
| $SPI_{10}$ | MOSI Setup time before SPCK falls | 3.5 | – | 3.4 | – | ns |
| $SPI_{11}$ | MOSI Hold time after SPCK falls | 0.8 | – | 0.7 | – | ns |
| $SPI_{12}$ | NPCS0 setup to SPCK rising | 4.9 | – | 4.8 | – | ns |
| $SPI_{13}$ | NPCS0 hold after SPCK falling | 1.1 | – | 0.9 | – | ns |
| $SPI_{14}$ | NPCS0 setup to SPCK falling | 4.4 | – | 4.4 | – | ns |
| $SPI_{15}$ | NPCS0 hold after SPCK rising | 0.5 | – | 0.3 | – | ns |
| $SPI_{16}$ | NPCS0 falling to MISO valid | 14.7 | – | 13.4 | – | ns |

**Table 66-63. SPI1 IOSET2 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Master Mode | | | | | |
| $SPI_0$ | MISO Setup time before SPCK rises | 15.5 | – | 13.6 | – | ns |

.....continued

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| $SPI_1$ | MISO Hold time after SPCK rises | 0 | – | 0 | – | ns |
| $SPI_2$ | SPCK rising to MOSI | 0 | 1.2 | 0 | 1.7 | ns |
| $SPI_3$ | MISO Setup time before SPCK falls | 14.9 | – | 13.7 | – | ns |
| $SPI_4$ | MISO Hold time after SPCK falls | 0 | – | 0 | – | ns |
| $SPI_5$ | SPCK falling to MOSI | 0 | 0.5 | 0 | 1.6 | ns |
| | Slave Mode | | | | | |
| $SPI_6$ | SPCK falling to MISO | 10.7 | 12.9 | 8.6 | 11.1 | ns |
| $SPI_7$ | MOSI Setup time before SPCK rises | 3 | – | 2.9 | – | ns |
| $SPI_8$ | MOSI Hold time after SPCK rises | 1 | – | 0.9 | – | ns |
| $SPI_9$ | SPCK rising to MISO | 10.3 | 12.4 | 8.2 | 10.5 | ns |
| $SPI_{10}$ | MOSI Setup time before SPCK falls | 3 | – | 2.9 | – | ns |
| $SPI_{11}$ | MOSI Hold time after SPCK falls | 1 | – | 0.9 | – | ns |
| $SPI_{12}$ | NPCS0 setup to SPCK rising | 4.4 | – | 4.3 | – | ns |
| $SPI_{13}$ | NPCS0 hold after SPCK falling | 1 | – | 0.9 | – | ns |
| $SPI_{14}$ | NPCS0 setup to SPCK falling | 4 | – | 4 | – | ns |
| $SPI_{15}$ | NPCS0 hold after SPCK rising | 0.4 | – | 0.3 | – | ns |
| $SPI_{16}$ | NPCS0 falling to MISO valid | 14.5 | – | 12.9 | – | ns |

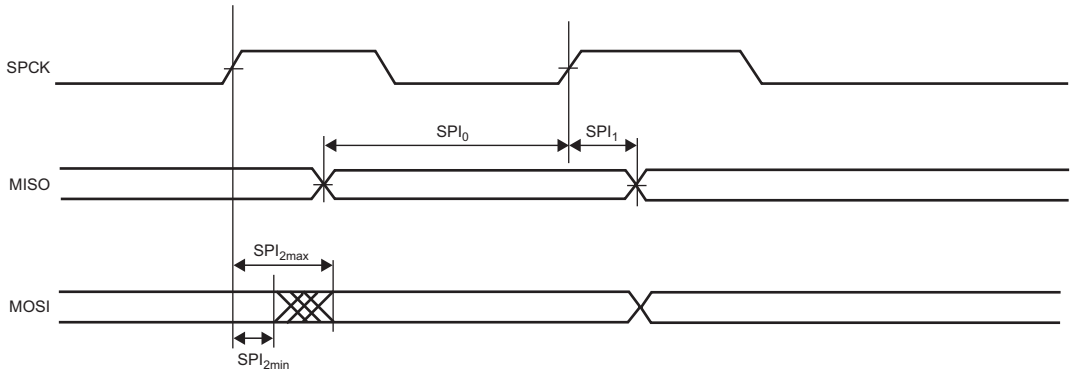
**Table 66-64. SPI1 IOSET3 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Master Mode | | | | | |
| $SPI_0$ | MISO Setup time before SPCK rises | 13.7 | – | 11.7 | – | ns |
| $SPI_1$ | MISO Hold time after SPCK rises | 0 | – | 0 | – | ns |
| $SPI_2$ | SPCK rising to MOSI | 0 | 3.1 | 0 | 2.9 | ns |
| $SPI_3$ | MISO Setup time before SPCK falls | 14.1 | – | 12.4 | – | ns |
| $SPI_4$ | MISO Hold time after SPCK falls | 0 | – | 0 | – | ns |
| $SPI_5$ | SPCK falling to MOSI | 0 | 3.1 | 0 | 3.3 | ns |
| | Slave Mode | | | | | |
| $SPI_6$ | SPCK falling to MISO | 9.5 | 11.4 | 7.5 | 9.6 | ns |
| $SPI_7$ | MOSI Setup time before SPCK rises | 4.5 | – | 4.4 | – | ns |
| $SPI_8$ | MOSI Hold time after SPCK rises | 0.7 | – | 0.5 | – | ns |
| $SPI_9$ | SPCK rising to MISO | 9.1 | 11 | 7.1 | 9.8 | ns |
| $SPI_{10}$ | MOSI Setup time before SPCK falls | 4.5 | – | 4.4 | – | ns |
| $SPI_{11}$ | MOSI Hold time after SPCK falls | 0.7 | – | 0.5 | – | ns |
| $SPI_{12}$ | NPCS0 setup to SPCK rising | 5.1 | – | 4.9 | – | ns |

.....continued

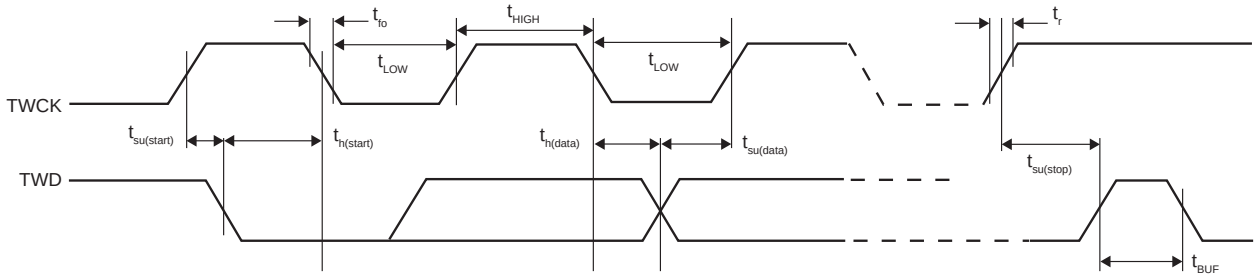
| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| $SPI_{13}$ | NPCS0 hold after SPCK falling | 0.9 | – | 0.8 | – | ns |
| $SPI_{14}$ | NPCS0 setup to SPCK falling | 4.7 | – | 4.6 | – | ns |
| $SPI_{15}$ | NPCS0 hold after SPCK rising | 0.3 | – | 0.2 | – | ns |
| $SPI_{16}$ | NPCS0 falling to MISO valid | 13.9 | – | 12.2 | – | ns |

**Figure 66-24. Minimum and Maximum Access Time for SPI Output Signal**




## 66.18 TWI Timings

**Figure 66-25. Two-wire Serial Bus Timing**




The table below describes the requirements for devices connected to the Two-wire Serial Bus.

**Table 66-65. Two-wire Serial Bus Requirements**

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $V_{IL}$ | Input Low-voltage | – | -0.3 | 0.3 × $V_{DDIO}$ | V |
| $V_{IH}$ | Input High-voltage | – | 0.7 × $V_{DDIO}$ | $V_{CC}$ + 0.3 | V |
| $V_{hys}$ | Hysteresis of Schmitt Trigger Inputs | – | 0.150 | – | V |
| $V_{OL}$ | Output Low-voltage | 3 mA sink current | – | 0.4 | V |
| $t_r$ | Rise Time for both TWD and TWCK | – | 20 + 0.1$C_b$ (2) | 300 | ns |
| $t_{fo}$ | Output Fall Time from $V_{IHmin}$ to $V_{ILmax}$ | 10 pF < $C_b$ < 400 pF (see the figure above) | 20 + 0.1$C_b$ (2) | 250 | ns |
| $C_i$ (1) | Capacitance for each I/O Pin | – | – | 10 | pF |
| $f_{TWCK}$ | TWCK Clock Frequency | – | 0 | 400 | kHz |

.....continued

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| Rp | Value of Pull-up Resistor | $f_{TWCK}$≤ 100 kHz | ($V_{DDIO}$ - 0.4V) ÷ 3mA | 1000ns ÷ $C_b$ | Ω |
| | | $f_{TWCK}$ > 100 kHz | ($V_{DDIO}$ - 0.4V) ÷ 3mA | 300ns ÷ $C_b$ | Ω |
| $t_{LOW}$ | Low Period of the TWCK Clock | $f_{TWCK}$≤ 100 kHz | (3) | – | µs |
| | | $f_{TWCK}$ > 100 kHz | (3) | – | µs |
| $t_{HIGH}$ | High Period of the TWCK Clock | $f_{TWCK}$≤ 100 kHz | (4) | – | µs |
| | | $f_{TWCK}$ > 100 kHz | (4) | – | µs |
| $t_{h(start)}$ | Hold Time (repeated) START condition | $f_{TWCK}$≤ 100 kHz | $t_{HIGH}$ | – | µs |
| | | $f_{TWCK}$ > 100 kHz | $t_{HIGH}$ | – | µs |
| $t_{su(start)}$ | Setup Time for a Repeated START condition | $f_{TWCK}$≤ 100 kHz | $t_{HIGH}$ | – | µs |
| | | $f_{TWCK}$ > 100 kHz | $t_{HIGH}$ | – | µs |
| $t_{h(data)}$ | Data Hold Time | $f_{TWCK}$≤ 100 kHz | 0 | (HOLD + 3) × $t_{peripheral\ clock}$ | µs |
| | | $f_{TWCK}$ > 100 kHz | 0 | (HOLD + 3) × $t_{peripheral\ clock}$ | µs |
| $t_{su(data)}$ | Data Setup Time | $f_{TWCK}$≤ 100 kHz | $t_{LOW}$ - (HOLD + 3) × $t_{peripheral\ clock}$ | – | ns |
| | | $f_{TWCK}$ > 100 kHz | $t_{LOW}$ - (HOLD + 3) × $t_{peripheral\ clock}$ | – | ns |
| $t_{su(stop)}$ | Setup time for STOP condition | $f_{TWCK}$≤ 100 kHz | $t_{HIGH}$ | – | µs |
| | | $f_{TWCK}$ > 100 kHz | $t_{HIGH}$ | – | µs |
| $t_{BUF}$ | Bus free time between a STOP and START condition | $f_{TWCK}$≤ 100 kHz | $t_{LOW}$ | – | µs |
| | | $f_{TWCK}$ > 100 kHz | $t_{LOW}$ | – | µs |

**Note:**

1. Required only for $f_{TWCK}$ > 100 kHz.
2. $C_B$ = capacitance of one bus line in pF. Per I2C Standard, $C_b$ Max = 400 pF
3. The TWCK low period is defined as follows: $t_{LOW} = ((CLDIV \times 2^{CKDIV}) + 3) \times t_{MCK}$
4. The TWCK high period is defined as follows: $t_{HIGH} = ((CHDIV \times 2^{CKDIV}) + 3) \times t_{MCK}$

## 66.19 QSPI Timings

### 66.19.1 Maximum QSPI Frequency

The following formulas give maximum QSPI frequency in Master Read and Write modes.

- Master Write Mode
  The QSPI sends data to a slave device only, e.g. an LCD. The limit is given by $QSPI_2$ (or $QSPI_5$) timing.
- Master Read Mode

$$f_{QSCK}\max = \frac{1}{\mathrm{QSPI}_0(\text{or } \mathrm{QSPI}_3) + t_{VALID}}$$

$t_{VALID}$ is the slave time response to output data after detecting a QSCK edge.

The $f_{QSCK}$ max is given between the maximum frequency given by the above formula and the pad I/O limitation.

### 66.19.2 Timing Conditions

Timings assuming a capacitance load are given in the table below.

**Table 66-66. Capacitance Load (QSPI 0 and QSPI1)**

| Supply | Corner | |
|---|---|---|
| | Max | Min |
| 3.3V | 30 pF | 5 pF |
| 1.8V | 20 pF | 5 pF |

### 66.19.3 Timing Extraction

**Figure 66-26. QSPI Master Mode 0**

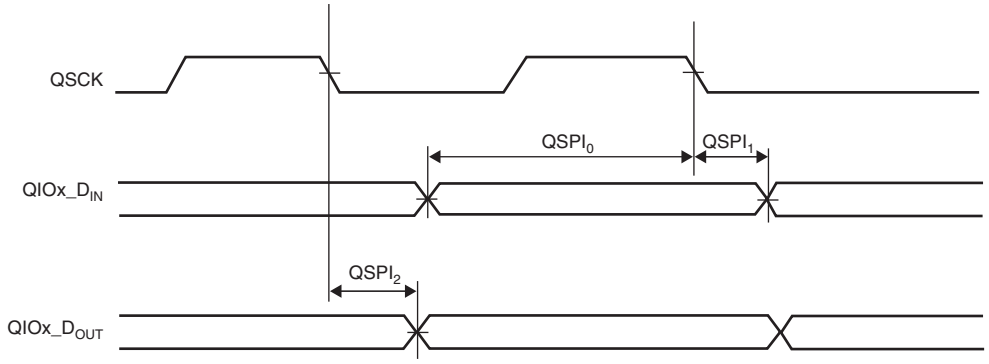



**Figure 66-27. QSPI Master Mode 1**

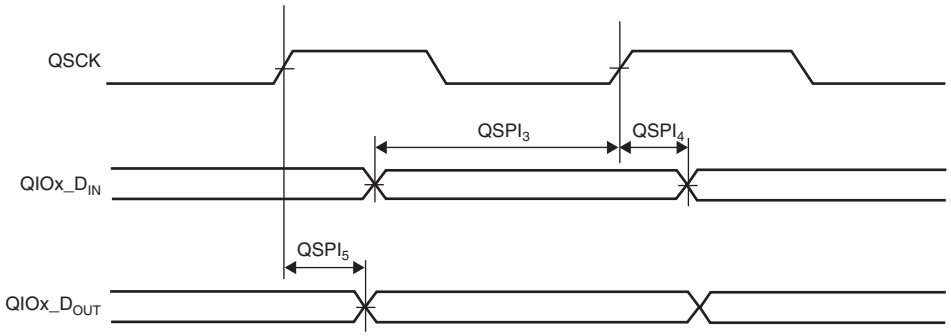



**Figure 66-28. QSPI Master Mode 2**

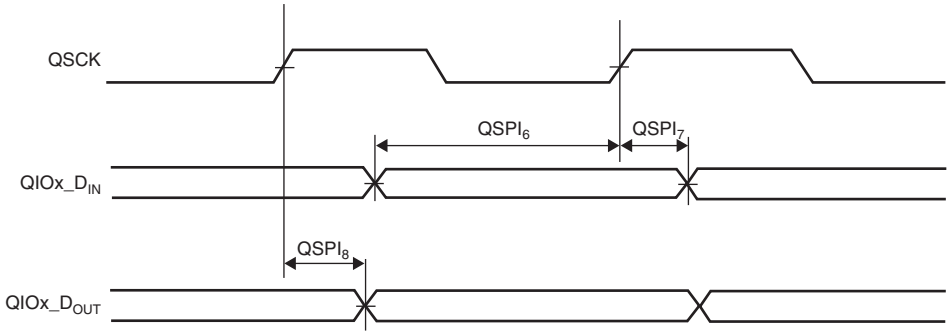


**Figure 66-29. QSPI Master Mode 3**

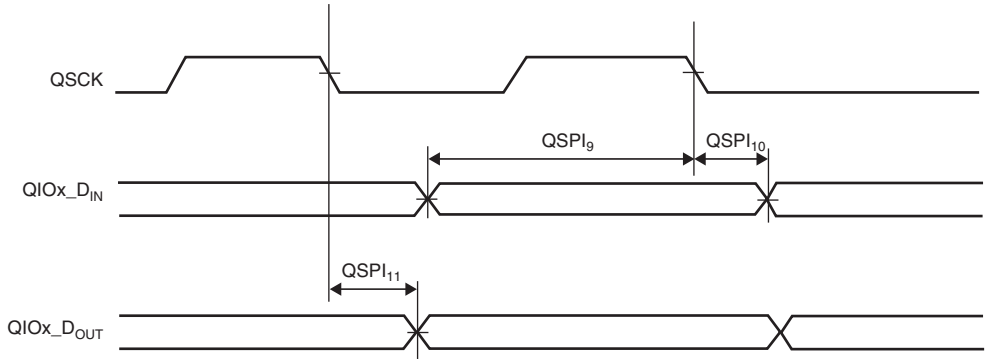



**Table 66-67. QSPI0 IOSET1 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Mode 0 | | | | | |
| $QSPI_0$ | QIOx Input setup time before SCK falls | 1.5 | – | 1.4 | – | ns |
| $QSPI_1$ | QIOx Input hold time after SCK falls | 0.4 | – | 0.5 | – | ns |
| $QSPI_2$ | SCK falling to QIOx valid | 0 | 3.4 | 0 | 2.4 | ns |
| | Mode 1 | | | | | |
| $QSPI_3$ | QIOx Input setup time before SCK rises | 11 | – | 8.7 | – | ns |
| $QSPI_4$ | QIOx Input hold time after SCK rises | 0.1 | – | 0 | – | ns |
| $QSPI_5$ | SCK rising to QIOx valid | 0 | 3 | 0 | 2.2 | ns |
| | Mode 2 | | | | | |
| $QSPI_6$ | QIOx Input setup time before SCK rises | 1.7 | – | 1.4 | – | ns |
| $QSPI_7$ | QIOx Input hold time after SCK rises | 0.1 | – | 0 | – | ns |
| $QSPI_8$ | SCK rising to QIOx valid | 0 | 3.3 | 0 | 2.3 | ns |
| | Mode 3 | | | | | |
| $QSPI_9$ | QIOx Input setup time before SCK falls | 11 | – | 8.9 | – | ns |
| $QSPI_{10}$ | QIOx Input hold time after SCK falls | 0.4 | – | 0.4 | – | ns |
| $QSPI_{11}$ | SCK falling to QIOx valid | 0 | 3.2 | 0 | 2.4 | ns |

**Table 66-68. QSPI0 IOSET2 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Mode 0 | | | | | |
| $QSPI_0$ | QIOx Input setup time before SCK falls | 2.2 | – | 2.2 | – | ns |
| $QSPI_1$ | QIOx Input hold time after SCK falls | 0.8 | – | 0.7 | – | ns |
| $QSPI_2$ | SCK falling to QIOx valid | 0 | 1.8 | 0 | 2 | ns |
| | Mode 1 | | | | | |
| $QSPI_3$ | QIOx Input setup time before SCK rises | 12.7 | – | 10.5 | – | ns |

.....continued

| Symbol | Parameter | Power Supply | 1.8V | | 3.3V | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| $QSPI_4$ | QIOx Input hold time after SCK rises | | 0.3 | – | 0.2 | – | ns |
| $QSPI_5$ | SCK rising to QIOx valid | | 0 | 2.2 | 0 | 1.9 | ns |
| Mode 2 | | | | | | | |
| $QSPI_6$ | QIOx Input setup time before SCK rises | | 2.6 | – | 2.5 | – | ns |
| $QSPI_7$ | QIOx Input hold time after SCK rises | | 0.3 | – | 0.2 | – | ns |
| $QSPI_8$ | SCK rising to QIOx valid | | 0 | 2.4 | 0 | 2 | ns |
| Mode 3 | | | | | | | |
| $QSPI_9$ | QIOx Input setup time before SCK falls | | 12 | – | 10.5 | – | ns |
| $QSPI_{10}$ | QIOx Input hold time after SCK falls | | 0.8 | – | 0.7 | – | ns |
| $QSPI_{11}$ | SCK falling to QIOx valid | | 0 | 1.5 | 0 | 1.9 | ns |

**Table 66-69. QSPI0 IOSET3 Timings**

| Symbol | Parameter | Power Supply | 1.8V | | 3.3V | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| Mode 0 | | | | | | | |
| $QSPI_0$ | QIOx Input setup time before SCK falls | | 1.8 | – | 1.7 | – | ns |
| $QSPI_1$ | QIOx Input hold time after SCK falls | | 0.8 | – | 0.7 | – | ns |
| $QSPI_2$ | SCK falling to QIOx valid | | 0 | 1.8 | 0 | 2.1 | ns |
| Mode 1 | | | | | | | |
| $QSPI_3$ | QIOx Input setup time before SCK rises | | 12.3 | – | 10.1 | – | ns |
| $QSPI_4$ | QIOx Input hold time after SCK rises | | 0.5 | – | 0.3 | – | ns |
| $QSPI_5$ | SCK rising to QIOx valid | | 0 | 2.2 | 0 | 2 | ns |
| Mode 2 | | | | | | | |
| $QSPI_6$ | QIOx Input setup time before SCK rises | | 2 | – | 1.7 | – | ns |
| $QSPI_7$ | QIOx Input hold time after SCK rises | | 0.5 | – | 0.3 | – | ns |
| $QSPI_8$ | SCK rising to QIOx valid | | 0 | 2.5 | 0 | 2.2 | ns |
| Mode 3 | | | | | | | |
| $QSPI_9$ | QIOx Input setup time before SCK falls | | 11.7 | – | 10.2 | – | ns |
| $QSPI_{10}$ | QIOx Input hold time after SCK falls | | 0.8 | – | 0.7 | – | ns |
| $QSPI_{11}$ | SCK falling to QIOx valid | | 0 | 1.5 | 1.2 | 2 | ns |

**Table 66-70. QSPI1 IOSET1 Timings**

| Symbol | Parameter | Power Supply | 1.8V | | 3.3V | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| Mode 0 | | | | | | | |
| $QSPI_0$ | QIOx Input setup time before SCK falls | | 1.1 | – | 0.9 | – | ns |

.....continued

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| $QSPI_1$ | QIOx Input hold time after SCK falls | 1 | – | 0.7 | – | ns |
| $QSPI_2$ | SCK falling to QIOx valid | 0 | 3.2 | 0 | 2.4 | ns |
| | Mode 1 | | | | | |
| $QSPI_3$ | QIOx Input setup time before SCK rises | 12 | – | 9.7 | – | ns |
| $QSPI_4$ | QIOx Input hold time after SCK rises | 0.8 | – | 0.5 | – | ns |
| $QSPI_5$ | SCK rising to QIOx valid | 0 | 2.7 | 0 | 2.1 | ns |
| | Mode 2 | | | | | |
| $QSPI_6$ | QIOx Input setup time before SCK rises | 1.1 | – | 0.8 | – | ns |
| $QSPI_7$ | QIOx Input hold time after SCK rises | 0.8 | – | 0.5 | – | ns |
| $QSPI_8$ | SCK rising to QIOx valid | 0 | 3 | 0 | 2.3 | ns |
| | Mode 3 | | | | | |
| $QSPI_9$ | QIOx Input setup time before SCK falls | 12.2 | – | 10 | – | ns |
| $QSPI_{10}$ | QIOx Input hold time after SCK falls | 1 | – | 0.7 | – | ns |
| $QSPI_{11}$ | SCK falling to QIOx valid | 0 | 3 | 0 | 2.4 | ns |

**Table 66-71. QSPI1 IOSET2 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Mode 0 | | | | | |
| $QSPI_0$ | QIOx Input setup time before SCK falls | 1.9 | – | 1.9 | – | ns |
| $QSPI_1$ | QIOx Input hold time after SCK falls | 0.8 | – | 0.7 | – | ns |
| $QSPI_2$ | SCK falling to QIOx valid | 0 | 1.5 | 0 | 1.9 | ns |
| | Mode 1 | | | | | |
| $QSPI_3$ | QIOx Input setup time before SCK rises | 12.6 | – | 10.4 | – | ns |
| $QSPI_4$ | QIOx Input hold time after SCK rises | 0.2 | – | 0.1 | – | ns |
| $QSPI_5$ | SCK rising to QIOx valid | 0 | 1.9 | 0 | 1.8 | ns |
| | Mode 2 | | | | | |
| $QSPI_6$ | QIOx Input setup time before SCK rises | 2.2 | – | 2.1 | – | ns |
| $QSPI_7$ | QIOx Input hold time after SCK rises | 0.2 | – | 0.1 | – | ns |
| $QSPI_8$ | SCK rising to QIOx valid | 0 | 2.2 | 0 | 2 | ns |
| | Mode 3 | | | | | |
| $QSPI_9$ | QIOx Input setup time before SCK falls | 11.9 | – | 10.4 | – | ns |
| $QSPI_{10}$ | QIOx Input hold time after SCK falls | 0.8 | – | 0.7 | – | ns |
| $QSPI_{11}$ | SCK falling to QIOx valid | 0 | 1.3 | 0 | 1.9 | ns |

**Table 66-72. QSPI1 IOSET3 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Mode 0 | | | | | |
| $QSPI_0$ | QIOx Input setup time before SCK falls | 1.8 | – | 1.8 | – | ns |
| $QSPI_1$ | QIOx Input hold time after SCK falls | 0.9 | – | 0.7 | – | ns |
| $QSPI_2$ | SCK falling to QIOx valid | 0 | 1.3 | 0 | 1.8 | ns |
| | Mode 1 | | | | | |
| $QSPI_3$ | QIOx Input setup time before SCK rises | 13.1 | – | 10.9 | – | ns |
| $QSPI_4$ | QIOx Input hold time after SCK rises | 0.4 | – | 0.2 | – | ns |
| $QSPI_5$ | SCK rising to QIOx valid | 0 | 1.7 | 0 | 1.7 | ns |
| | Mode 2 | | | | | |
| $QSPI_6$ | QIOx Input setup time before SCK rises | 2.2 | – | 2.1 | – | ns |
| $QSPI_7$ | QIOx Input hold time after SCK rises | 0.4 | – | 0.2 | – | ns |
| $QSPI_8$ | SCK rising to QIOx valid | 0 | 2 | 0 | 1.8 | ns |
| | Mode 3 | | | | | |
| $QSPI_9$ | QIOx Input setup time before SCK falls | 12.5 | – | 11 | – | ns |
| $QSPI_{10}$ | QIOx Input hold time after SCK falls | 0.9 | – | 0.7 | – | ns |
| $QSPI_{11}$ | SCK falling to QIOx valid | 0 | 1.1 | 0 | 1.7 | ns |

## 66.20 MPDDRC Timings

The following characteristics are applicable to the ambient operating temperature range $T_A$ = -40°C to +105°C for -CN devices, and -40°C to +85°C for -CU devices, unless otherwise specified.

### 66.20.1 Board Design Constraints

As the SAMA5D2 series embeds impedance calibrated pads, there are no capacitive constraints on DDR signals. However, a board must be designed and equipped in order to respect propagation time and intrinsic delay in the SDRAM device. In all cases, line length to memory device must not exceed 5 cm.

### 66.20.2 DDR2-SDRAM

**Note:** For DDR2 memory, the SHIFT_SAMPLING field value in the MPRDDRC_RD_DATA_PATH register must be configured to 1.

**Table 66-73. System Clock Waveform Parameters**

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $t_{DDRCK}$ | DDRCK Cycle Time | VDDCORE[1.1V, 1.32V] | 7.5 | 8.0 | ns |
| | | VDDCORE[1.2V, 1.32V], VDDIODDR[1.8V, 1.9V] | 6.0 | 8.0 | ns |

### 66.20.3 LPDDR1-SDRAM

**Note:** For LPDDR1 memory, the SHIFT_SAMPLING field value in the MPRDDRC_RD_DATA_PATH register must be configured as follows:

SHIFT_SAMPLING = 0 for 0 < DDR_CLK < 94 MHz

SHIFT_SAMPLING = 1 for 94 MHz < DDR_CLK < 166 MHz

**Table 66-74. System Clock Waveform Parameters**

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $t_{DDRCK}$ | DDRCK Cycle Time | VDDCORE[1.1V, 1.32V] | 7.5 | – | ns |
| $t_{DDRCK}$ | DDRCK Cycle Time | VDDCORE[1.2V, 1.32V], VDDIODDR[1.8V, 1.9V] | 6 | – | ns |

### 66.20.4 LPDDR2/LPDDR3-SDRAM

**Note:** For LPDDR2/LPDDR3 memory, the SHIFT_SAMPLING field value in the MPRDDRC_RD_DATA_PATH register must be configured as follows:

SHIFT_SAMPLING = 0 for 0 < DDR_CLK < 80 MHz

SHIFT_SAMPLING = 1 for 80 MHz < DDR_CLK < 166 MHz

**Table 66-75. System Clock Waveform Parameters**

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $t_{DDRCK}$ | DDRCK Cycle Time | VDDCORE[1.1V, 1.32V] | 7.5 | – | ns |
| $t_{DDRCK}$ | DDRCK Cycle Time | VDDCORE[1.1V, 1.32V], VDDIODDR[1.18V, 1.3V] | 6 | – | ns |

### 66.20.5 DDR3/DDR3L-SDRAM

**Note:** For DDR3/DDR3L memory, the SHIFT_SAMPLING field value in the MPRDDRC_RD_DATA_PATH register must be configured to 2.

**Table 66-76. System Clock Waveform Parameters**

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $t_{DDRCK}$ | DDRCK Cycle Time | VDDCORE[1.1V, 1.32V] | 8.0 | – | ns |
| $t_{DDRCK}$ | DDRCK Cycle Time | VDDCORE[1.1V, 1.32V], VDDIODDR[1.18V, 1.3V] | 8.0 | – | ns |

## 66.21 SSC Timings

### 66.21.1 Timing Conditions

Timings assuming a capacitance load are given in the table below.

**Table 66-77. Capacitance Load (SSC0 and SSC1)**

| Supply | Corner | |
|---|---|---|
| | Max | Min |
| 3.3V | 30 pF | 5 pF |
| 1.8V | 20 pF | 5 pF |

### 66.21.2 Timing Extraction

**Figure 66-30. SSC Transmitter, TK and TF in Output**

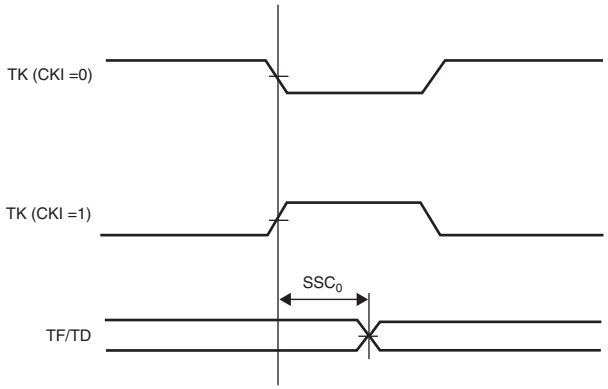



**Figure 66-31. SSC Transmitter, TK in Input and TF in Output**

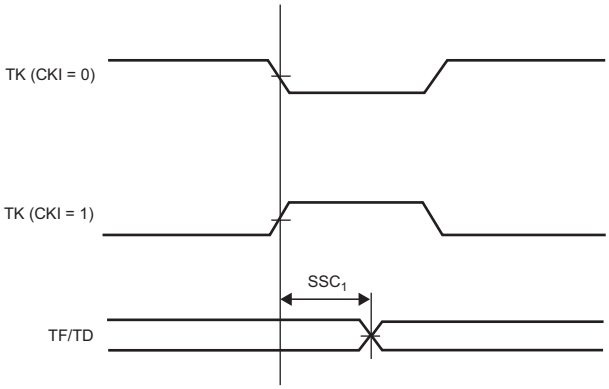



**Figure 66-32. SSC Transmitter, TK in Output and TF in Input**

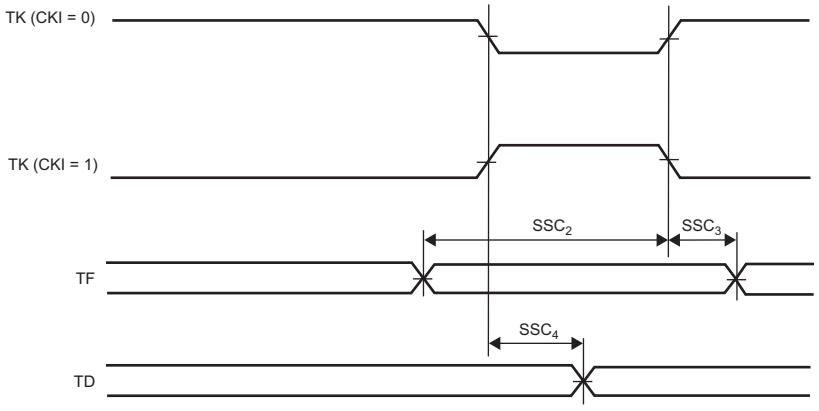


**Figure 66-33. SSC Transmitter, TK and TF in Input**

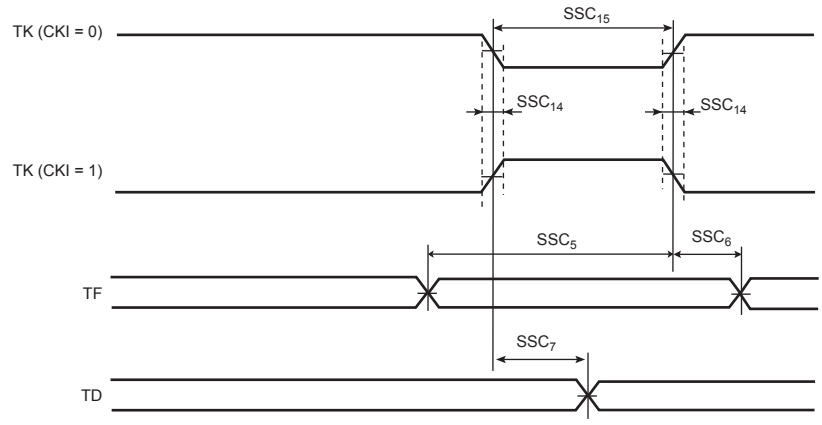



**Figure 66-34. SSC Receiver RK and RF in Input**

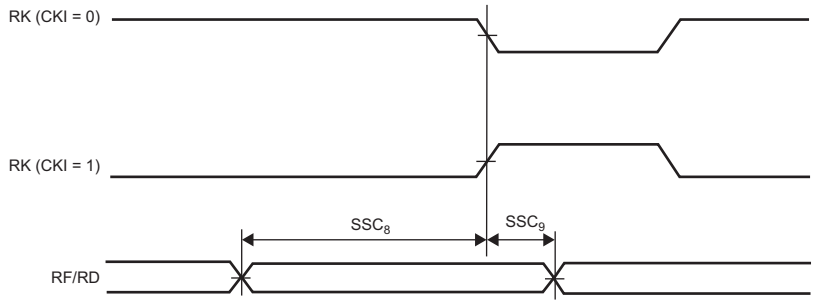



**Figure 66-35. SSC Receiver, RK in Input and RF in Output**

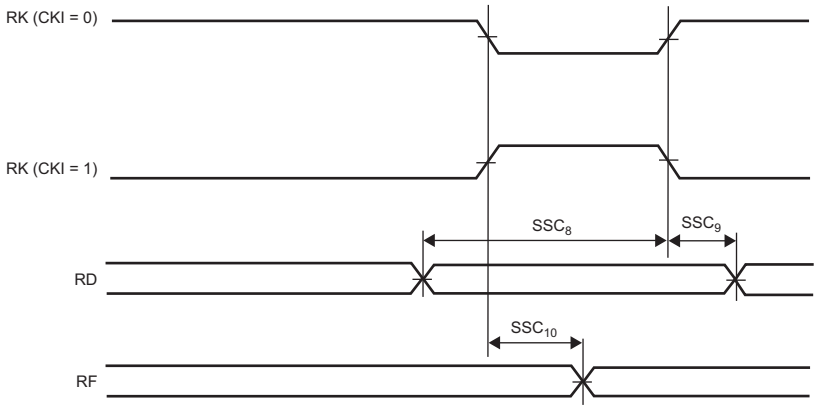


**Figure 66-36. SSC Receiver, RK and RF in Output**

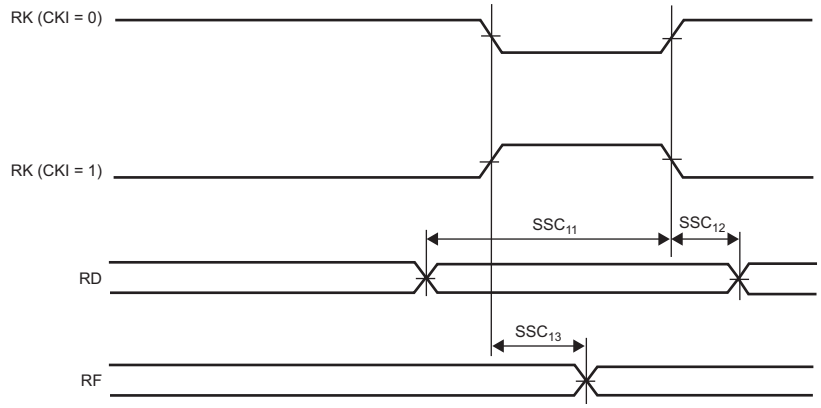



**Figure 66-37. SSC Receiver, RK in Output and RF in Input**

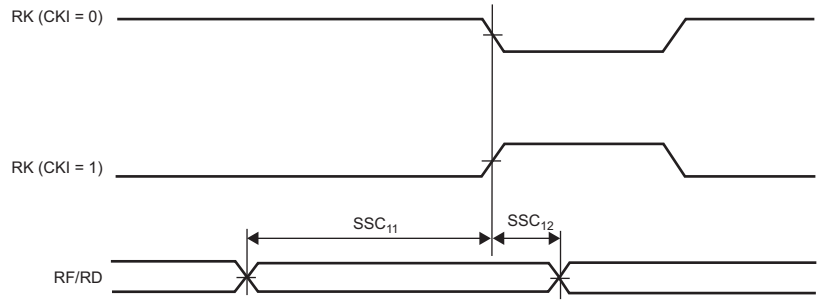



**Table 66-78. SSC0 IOSET1 Timings**

| Symbol | Power Supply / Parameter | Conditions | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|---|
| Transmitter | | | | | | | |
| $SSC_0$ | TK edge to TF/TD (TK output, TF output) | – | 0 | 3 | 0 | 3.3 | ns |
| $SSC_1$ | TK edge to TF/TD (TK input, TF output) | – | 3.7 | 13 | 3 | 11.3 | ns |
| $SSC_2$ | TF setup time before TK edge (TK output) | – | 12.8 | – | 11.2 | – | ns |
| $SSC_3$ | TF hold time after TK edge (TK output) | – | 0 | – | 0 | – | ns |
| $SSC_4$ | TK edge to TF/TD (TK output, TF input) | – | 0 | 3 | 0 | 3.3 | ns |
| | | STTDLY = 0<br>START = 4, 5 or 7 | 2 × $t_{CPMCK}$ | 3<br>+ (2 × $t_{CPMCK}$) | 2 × $t_{CPMCK}$ | 3.3<br>+ (2 × $t_{CPMCK}$) | ns |
| $SSC_5$ | TF setup time before TK edge (TK input) | – | 0 | – | 0 | – | ns |
| $SSC_6$ | TF hold time after TK edge (TK input) | – | $t_{CPMCK}$ | – | $t_{CPMCK}$ | – | ns |
| $SSC_7$ | TK edge to TF/TD (TK input, TF input) | – | 3.7 | 13 | 3.2 | 11.3 | ns |
| | | STTDLY = 0<br>START = 4, 5 or 7 | 3.7<br>+ (3 × $t_{CPMCK}$) | 13<br>+ (3 × $t_{CPMCK}$) | 3.2<br>+ (3 × $t_{CPMCK}$) | 11.3<br>+ (3 × $t_{CPMCK}$) | ns |
| Receiver | | | | | | | |

.....continued

| Symbol | Power Supply / Parameter | Conditions | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|---|
| $SSC_8$ | RF/RD setup time before RK edge (RK input) | – | 0 | – | 0 | – | ns |
| $SSC_9$ | RF/RD hold time after RK edge (RK input) | – | $t_{CPMCK}$ | – | $t_{CPMCK}$ | – | ns |
| $SSC_{10}$ | RK edge to RF (RK input) | – | 3.5 | 12 | 2.9 | 10.4 | ns |
| $SSC_{11}$ | RF/RD setup time before RK edge (RK output) | – | 13.6 - $t_{CPMCK}$ | – | 12 - $t_{CPMCK}$ | – | ns |
| $SSC_{12}$ | RF/RD hold time after RK edge (RK output) | – | $t_{CPMCK}$ | – | $t_{CPMCK}$ | – | ns |
| $SSC_{13}$ | RK edge to RF (RK output) | – | 0 | 3 | 0 | 3.3 | ns |
| $SSC_{14}$(1) | TK rise time or fall time | 10 to 90% | – | 10 | – | 10 | ns |
| $SSC_{15}$(1) | TK low or high time | $V_{TK}>V_{IH}$ or $V_{TK}<V_{IL}$ | 3.6 + (3 x $t_{CPMCK}$) | – | 3.4 + (3 x $t_{CPMCK}$) | – | ns |

**Note:**

1. Timings SSC14 and SSC15 also apply when RK is used instead of TK (SSC_TCMR.CKS = RK).

**Table 66-79. SSC0 IOSET2 Timings**

| Symbol | Power supply / Parameter | Conditions | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|---|
| Transmitter | | | | | | | |
| $SSC_0$ | TK edge to TF/TD (TK output, TF output) | – | 0 | 3.4 | 0 | 3.7 | ns |
| $SSC_1$ | TK edge to TF/TD (TK input, TF output) | – | 3.5 | 12.3 | 2.8 | 10.5 | ns |
| $SSC_2$ | TF setup time before TK edge (TK output) | – | 12 | – | 10.3 | – | ns |
| $SSC_3$ | TF hold time after TK edge (TK output) | – | 0 | – | 0 | – | ns |
| $SSC_4$ | TK edge to TF/TD (TK output, TF input) | – | 0 | 3.4 | 0 | 3.5 | ns |
| | | STTDLY = 0<br>START = 4, 5 or 7 | 2 × $t_{CPMCK}$ | 3.4<br>+ (2 × $t_{CPMCK}$) | 2 × $t_{CPMCK}$ | 3.5<br>+ (2 × $t_{CPMCK}$) | ns |
| $SSC_5$ | TF setup time before TK edge (TK input) | – | 0 | – | 0 | – | |
| $SSC_6$ | TF hold time after TK edge (TK input) | – | $t_{CPMCK}$ | – | $t_{CPMCK}$ | – | |
| $SSC_7$ | TK edge to TF/TD (TK input, TF input) | – | 3.6 | 12.3 | 3 | 10.4 | |
| | | STTDLY = 0<br>START = 4, 5 or 7 | 3.6<br>+ (3 × $t_{CPMCK}$) | 12.3<br>+ (3 × $t_{CPMCK}$) | 3<br>+ (3 × $t_{CPMCK}$) | 10.4<br>+ (3 × $t_{CPMCK}$) | |
| Receiver | | | | | | | |
| $SSC_8$ | RF/RD setup time before RK edge (RK input) | – | 0 | – | 0 | – | ns |

.....continued

| Symbol | Power supply / Parameter | Conditions | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|---|
| $SSC_9$ | RF/RD hold time after RK edge (RK input) | – | $t_{CPMCK}$ | – | $t_{CPMCK}$ | – | ns |
| $SSC_{10}$ | RK edge to RF (RK input) | – | 3.3 | 11.5 | 2.7 | 9.8 | ns |
| $SSC_{11}$ | RF/RD setup time before RK edge (RK output) | – | 13.8 - $t_{CPMCK}$ | – | 12.1 - $t_{CPMCK}$ | – | ns |
| $SSC_{12}$ | RF/RD hold time after RK edge (RK output) | – | $t_{CPMCK}$ | – | $t_{CPMCK}$ | – | ns |
| $SSC_{13}$ | RK edge to RF (RK output) | – | 0 | 2.8 | 0 | 3.1 | ns |
| $SSC_{14}$(1) | TK rise time or fall time | 10 to 90% | – | 10 | – | 10 | ns |
| $SSC_{15}$(1) | TK low or high time | $V_{TK}>V_{IH}$ or $V_{TK}<V_{IL}$ | 3.6 + (3 x $t_{CPMCK}$) | – | 3.4 + (3 x $t_{CPMCK}$) | – | ns |

**Note:**

1. Timings SSC14 and SSC15 also apply when RK is used instead of TK (SSC_TCMR.CKS = RK).

**Table 66-80. SSC1 IOSET1 Timings**

| Symbol | Power supply / Parameter | Conditions | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|---|
| | | Transmitter | | | | | |
| $SSC_0$ | TK edge to TF/TD (TK output, TF output) | – | 0 | 2.6 | 0 | 2.7 | ns |
| $SSC_1$ | TK edge to TF/TD (TK input, TF output) | – | 3.6 | 12.7 | 3 | 10.9 | ns |
| $SSC_2$ | TF setup time before TK edge (TK output) | – | 13.4 | – | 11.2 | – | ns |
| $SSC_3$ | TF hold time after TK edge (TK output) | – | 0 | – | 0 | – | ns |
| $SSC_4$ | TK edge to TF/TD (TK output, TF input) | – | 0 | 2.1 | 0 | 2 | ns |
| | | STTDLY = 0<br>START = 4, 5 or 7 | 2 × $t_{CPMCK}$ | 2.1<br>+ (2 × $t_{CPMCK}$) | 2 × $t_{CPMCK}$ | 2<br>+ (2 × $t_{CPMCK}$) | ns |
| $SSC_5$ | TF setup time before TK edge (TK input) | – | 0 | – | 0 | – | |
| $SSC_6$ | TF hold time after TK edge (TK input) | – | $t_{CPMCK}$ | – | $t_{CPMCK}$ | – | |
| $SSC_7$ | TK edge to TF/TD (TK input, TF input) | – | 3.6 | 12.2 | 3 | 10.2 | |
| | | STTDLY = 0<br>START = 4, 5 or 7 | 3.6<br>+ (3 × $t_{CPMCK}$) | 12.2<br>+ (3 × $t_{CPMCK}$) | 3<br>+ (3 × $t_{CPMCK}$) | 10.2<br>+ (3 × $t_{CPMCK}$) | |
| | | Receiver | | | | | |
| $SSC_8$ | RF/RD setup time before RK edge (RK input) | – | 0 | – | 0 | – | ns |
| $SSC_9$ | RF/RD hold time after RK edge (RK input) | – | $t_{CPMCK}$ | – | $t_{CPMCK}$ | – | ns |

.....continued

| Symbol | Power supply / Parameter | Conditions | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|---|
| $SSC_{10}$ | RK edge to RF (RK input) | – | 3.4 | 11.8 | 2.7 | 9.9 | ns |
| $SSC_{11}$ | RF/RD setup time before RK edge (RK output) | – | 12.2 - $t_{CPMCK}$ | – | 10.3 - $t_{CPMCK}$ | – | ns |
| $SSC_{12}$ | RF/RD hold time after RK edge (RK output) | – | $t_{CPMCK}$ | – | $t_{CPMCK}$ | – | ns |
| $SSC_{13}$ | RK edge to RF (RK output) | – | 0 | 3.3 | 0 | 3.4 | ns |
| $SSC_{14}$(1) | TK rise time or fall time | 10 to 90% | – | 10 | – | 10 | ns |
| $SSC_{15}$(1) | TK low or high time | $V_{TK}>V_{IH}$ or $V_{TK}<V_{IL}$ | 3.6 + (3 x $t_{CPMCK}$) | – | 3.4 + (3 x $t_{CPMCK}$) | – | ns |

**Note:**

1. Timings SSC14 and SSC15 also apply when RK is used instead of TK (SSC_TCMR.CKS = RK).

**Table 66-81. SSC1 IOSET2 Timings**

| Symbol | Power supply / Parameter | Conditions | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|---|
| Transmitter | | | | | | | |
| $SSC_0$ | TK edge to TF/TD (TK output, TF output) | – | 0 | 2.5 | 0 | 2.6 | ns |
| $SSC_1$ | TK edge to TF/TD (TK input, TF output) | – | 3.7 | 13 | 3.1 | 11.3 | ns |
| $SSC_2$ | TF setup time before TK edge (TK output) | – | 14.3 | – | 12.2 | – | ns |
| $SSC_3$ | TF hold time after TK edge (TK output) | – | 0 | – | 0 | – | ns |
| $SSC_4$ | TK edge to TF/TD (TK output, TF input) | – | 0 | 2.1 | 0 | 1.8 | ns |
| | | STTDLY = 0<br>START = 4, 5 or 7 | 2 × $t_{CPMCK}$ | 2.1<br>+ (2 × $t_{CPMCK}$) | 2 × $t_{CPMCK}$ | 1.8<br>+ (2 × $t_{CPMCK}$) | ns |
| $SSC_5$ | TF setup time before TK edge (TK input) | – | 0 | – | 0 | – | |
| $SSC_6$ | TF hold time after TK edge (TK input) | – | $t_{CPMCK}$ | – | $t_{CPMCK}$ | – | |
| $SSC_7$ | TK edge to TF/TD (TK input, TF input) | – | 3.7 | 12.6 | 3.1 | 10.4 | |
| | | STTDLY = 0<br>START = 4, 5 or 7 | 3.7<br>+ (3 × $t_{CPMCK}$) | 12.6<br>+ (3 × $t_{CPMCK}$) | 3.1<br>+ (3 × $t_{CPMCK}$) | 10.4<br>+ (3 × $t_{CPMCK}$) | |
| Receiver | | | | | | | |
| $SSC_8$ | RF/RD setup time before RK edge (RK input) | – | 0 | – | 0 | – | ns |
| $SSC_9$ | RF/RD hold time after RK edge (RK input) | – | $t_{CPMCK}$ | – | $t_{CPMCK}$ | – | ns |
| $SSC_{10}$ | RK edge to RF (RK input) | – | 3.6 | 12.2 | 3 | 10.3 | ns |

.....continued

| Symbol | Power supply<br>Parameter | Conditions | 1.8V<br>Min | 1.8V<br>Max | 3.3V<br>Min | 3.3V<br>Max | Unit |
|---|---|---|---|---|---|---|---|
| $SSC_{11}$ | RF/RD setup time before RK edge (RK output) | – | 12.3 - $t_{CPMCK}$ | – | 10.5 - $t_{CPMCK}$ | – | ns |
| $SSC_{12}$ | RF/RD hold time after RK edge (RK output) | – | $t_{CPMCK}$ | – | $t_{CPMCK}$ | – | ns |
| $SSC_{13}$ | RK edge to RF (RK output) | – | 0 | 2.9 | 0 | 3.1 | ns |
| $SSC_{14}$(1) | TK rise time or fall time | 10 to 90% | – | 10 | – | 10 | ns |
| $SSC_{15}$(1) | TK low or high time | $V_{TK}>V_{IH}$ or $V_{TK}<V_{IL}$ | 3.6 + (3 x $t_{CPMCK}$) | – | 3.4 + (3 x $t_{CPMCK}$) | – | ns |

**Note:**

1. Timings SSC14 and SSC15 also apply when RK is used instead of TK (SSC_TCMR.CKS = RK).

## 66.22 PDMIC Timings

### 66.22.1 Timing Conditions

Timings assuming capacitance loads are given in the table below.

**Table 66-82. Capacitance Load**

| Supply | Corner<br>Max | Corner<br>Min |
|---|---|---|
| 3.3V | 30 pF | 5 pF |
| 1.8V | 20 pF | 5 pF |

### 66.22.2 Timing Extraction

**Figure 66-38. PDMIC Timing Diagram**

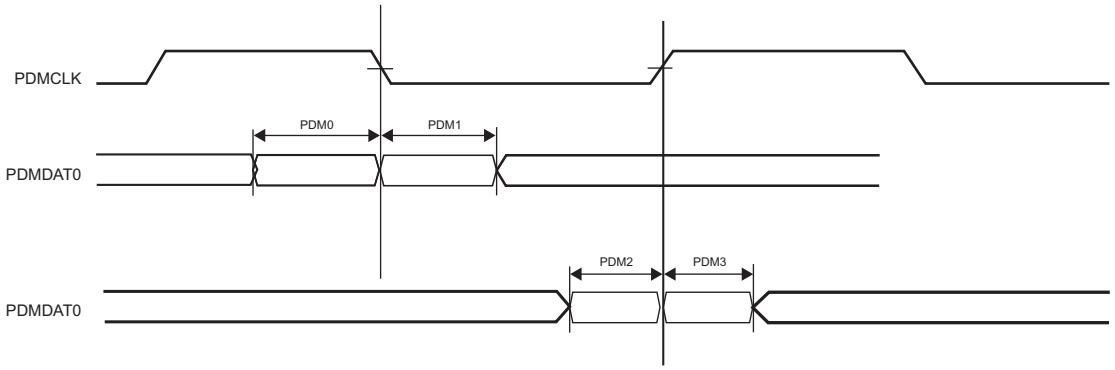



**Table 66-83. PDMIC IOSET1 Timings**

| Symbol | Power Supply<br>Parameter | 1.8V<br>Min | 1.8V<br>Max | 3.3V<br>Min | 3.3V<br>Max | Unit |
|---|---|---|---|---|---|---|
| $PDMIC_0$ | DATA setup time right | 3.5 | – | 3.5 | – | ns |
| $PDMIC_1$ | DATA hold time right | 3.1 | – | 3.5 | – | ns |

.....continued

| Symbol | Parameter | Power Supply | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|---|
| $PDMIC_2$ | DATA setup time left | | 3.5 | – | 3.5 | – | ns |
| $PDMIC_3$ | DATA hold time left | | 3.1 | – | 3.5 | – | ns |

**Table 66-84. PDMIC IOSET2 Timings**

| Symbol | Parameter | Power Supply | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|---|
| $PDMIC_0$ | DATA setup time right | | 4.2 | – | 4.2 | – | ns |
| $PDMIC_1$ | DATA hold time right | | 2 | – | 2 | – | ns |
| $PDMIC_2$ | DATA setup time left | | 4.2 | – | 4.2 | – | ns |
| $PDMIC_3$ | DATA hold time left | | 2 | – | 2 | – | ns |

## 66.23 I2SC Timings

### 66.23.1 Timing Conditions

Timings assuming capacitance loads are given in the table below.

**Table 66-85. Capacitance Load (I2SC0 and I2SC1)**

| Supply | Corner Max | Corner Min |
|---|---|---|
| 3.3V | 30 pF | 5 pF |
| 1.8V | 20 pF | 5 pF |

### 66.23.2 Timing Extraction

**Table 66-86. I2SC0 IOSET1 Timings**

| Symbol | Parameter | Power Supply | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|---|
| | Master | | | | | | |
| $I2SC_0$ | SDI Input setup time before SCK rises | | 12.5 | – | 10.8 | – | ns |
| $I2SC_1$ | SDI Input hold time after SCK rises | | 0 | – | 0 | – | ns |
| $I2SC_2$ | SCK falling to SDO valid | | 0 | 3.9 | 0 | 4 | ns |
| $I2SC_3$ | SCK falling to WS valid | | 0 | 2.7 | 0 | 3.1 | ns |
| | Slave | | | | | | |
| $I2SC_4$ | SDI Input setup time before SCK rises | | 1.1 | – | 1 | – | ns |
| $I2SC_5$ | SDI Input hold time after SCK rises | | 1.3 | – | 1.2 | – | ns |
| $I2SC_6$ | WS Input setup time before SCK rises | | 2 | – | 1.8 | – | ns |
| $I2SC_7$ | WS Input hold time after SCK rises | | 0.9 | – | 0.8 | – | ns |
| $I2SC_8$ | SCK falling to SDO valid | | 4.2 | 14 | 3.6 | 12 | ns |

**Table 66-87. I2SC0 IOSET2 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Master | | | | | |
| $I2SC_0$ | SDI Input setup time before SCK rises | 11.7 | – | 9.7 | – | ns |
| $I2SC_1$ | SDI Input hold time after SCK rises | 0 | – | 0 | – | ns |
| $I2SC_2$ | SCK falling to SDO valid | 0 | 3 | 0 | 3 | ns |
| $I2SC_3$ | SCK falling to WS valid | 0.1 | 4.7 | 0.2 | 4.8 | ns |
| | Slave | | | | | |
| $I2SC_4$ | SDI Input setup time before SCK rises | 1.7 | – | 1.5 | – | ns |
| $I2SC_5$ | SDI Input hold time after SCK rises | 0.6 | – | 0.4 | – | ns |
| $I2SC_6$ | WS Input setup time before SCK rises | 3.6 | – | 3.4 | – | ns |
| $I2SC_7$ | WS Input hold time after SCK rises | 0.7 | – | 0.6 | – | ns |
| $I2SC_8$ | SCK falling to SDO valid | 3.7 | 12 | 3 | 10 | ns |

**Table 66-88. I2SC1 IOSET1 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Master | | | | | |
| $I2SC_0$ | SDI Input setup time before SCK rises | 13.1 | – | 11.4 | – | ns |
| $I2SC_1$ | SDI Input hold time after SCK rises | 0 | – | 0 | – | ns |
| $I2SC_2$ | SCK falling to SDO valid | 0 | 3.5 | 0 | 3.6 | ns |
| $I2SC_3$ | SCK falling to WS valid | 0 | 2.9 | 0 | 3 | ns |
| | Slave | | | | | |
| $I2SC_4$ | SDI Input setup time before SCK rises | 1.4 | – | 1.3 | – | ns |
| $I2SC_5$ | SDI Input hold time after SCK rises | 1 | – | 0.8 | – | ns |
| $I2SC_6$ | WS Input setup time before SCK rises | 2.4 | – | 2.1 | – | ns |
| $I2SC_7$ | WS Input hold time after SCK rises | 0.8 | – | 0.7 | – | ns |
| $I2SC_8$ | SCK falling to SDO valid | 4.4 | 13.8 | 3.7 | 11.9 | ns |

**Table 66-89. I2SC1 IOSET2 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| | Master | | | | | |
| $I2SC_0$ | SDI Input setup time before SCK rises | 12.9 | – | 11.2 | – | ns |
| $I2SC_1$ | SDI Input hold time after SCK rises | 0 | – | 0 | – | ns |
| $I2SC_2$ | SCK falling to SDO valid | 0 | 3.6 | 0 | 3.7 | ns |
| $I2SC_3$ | SCK falling to WS valid | 0 | 2.9 | 0 | 3 | ns |
| | Slave | | | | | |

.....continued

| Symbol | Parameter | Power Supply | 1.8V | | 3.3V | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| $I2SC_4$ | SDI Input setup time before SCK rises | | 1.1 | – | 1 | – | ns |
| $I2SC_5$ | SDI Input hold time after SCK rises | | 1.2 | – | 1 | – | ns |
| $I2SC_6$ | WS Input setup time before SCK rises | | 2.2 | – | 2 | – | ns |
| $I2SC_7$ | WS Input hold time after SCK rises | | 0.9 | – | 0.7 | – | ns |
| $I2SC_8$ | SCK falling to SDO valid | | 4.3 | 14 | 3.7 | 12 | ns |

## 66.24 ISC Timings

### 66.24.1 Timing Conditions

Timings assuming capacitance loads are given in the table below.

**Table 66-90. Capacitance Load**

| Supply | Corner | |
|---|---|---|
| | Max | Min |
| 3.3V | 30 pF | 5 pF |
| 1.8V | 20 pF | 5 pF |

### 66.24.2 Timing Extraction

**Figure 66-39. ISC Timing Diagram**

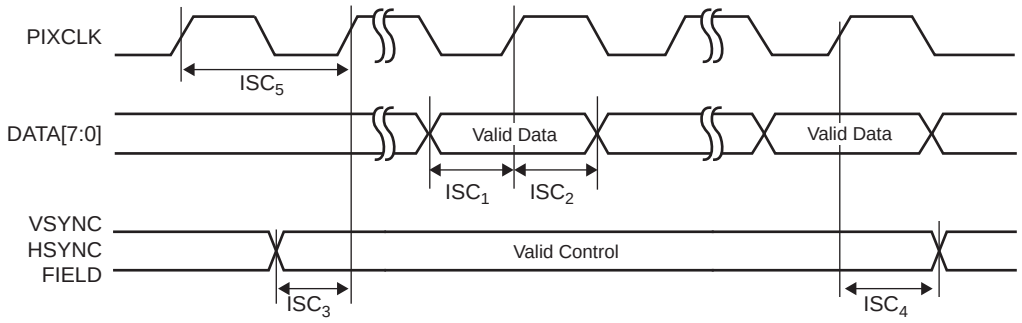



**Table 66-91. ISC IOSET1 Timings**

| Symbol | Parameter | Power Supply | 1.8V | | 3.3V | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| $ISC_1$ | DATA setup time before PIXCLK rises | | 3.9 | – | 3.6 | – | ns |
| $ISC_2$ | DATA hold time after PIXCLK rises | | 0.7 | – | 0.6 | – | ns |
| $ISC_3$ | VSYNC/HSYNC/FIELD setup time before PIXCLK rises | | 4.4 | – | 4.2 | – | ns |
| $ISC_4$ | CONTROL VSYNC/HSYNC/FIELD hold time after PIXCLK rises | | 0.4 | – | 0.3 | – | ns |
| $ISC_5$ | PIXCLK frequency | | – | 96 | – | 96 | MHz |

**Table 66-92. ISC IOSET2 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| $ISC_1$ | DATA setup time before PIXCLK rises | 4.3 | – | 4.2 | – | ns |
| $ISC_2$ | DATA hold time after PIXCLK rises | 0.5 | – | 0.3 | – | ns |
| $ISC_3$ | VSYNC/HSYNC/FIELD setup time before PIXCLK rises | 4.6 | – | 4.4 | – | ns |
| $ISC_4$ | CONTROL VSYNC/HSYNC/FIELD hold time after PIXCLK rises | 0.2 | – | 0 | – | ns |
| $ISC_5$ | PIXCLK frequency | – | 96 | – | 96 | MHz |

**Table 66-93. ISC IOSET3 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| $ISC_1$ | DATA setup time before PIXCLK rises | 4.6 | – | 4.2 | – | ns |
| $ISC_2$ | DATA hold time after PIXCLK rises | 0.5 | – | 0.4 | – | ns |
| $ISC_3$ | VSYNC/HSYNC/FIELD setup time before PIXCLK rises | 4.3 | – | 4 | – | ns |
| $ISC_4$ | CONTROL VSYNC/HSYNC/FIELD hold time after PIXCLK rises | 1.5 | – | 0.4 | – | ns |
| $ISC_5$ | PIXCLK frequency | – | 96 | – | 96 | MHz |

**Table 66-94. ISC IOSET4 Timings**

| Symbol | Power Supply / Parameter | 1.8V Min | 1.8V Max | 3.3V Min | 3.3V Max | Unit |
|---|---|---|---|---|---|---|
| $ISC_1$ | DATA setup time before PIXCLK rises | 4.3 | – | 4 | – | ns |
| $ISC_2$ | DATA hold time after PIXCLK rises | 0.5 | – | 0.4 | – | ns |
| $ISC_3$ | VSYNC/HSYNC/FIELD setup time before PIXCLK rises | 4.2 | – | 4 | – | ns |
| $ISC_4$ | CONTROL VSYNC/HSYNC/FIELD hold time after PIXCLK rises | 0.5 | – | 0.3 | – | ns |
| $ISC_5$ | PIXCLK frequency | – | 96 | – | 96 | MHz |

## 66.25 SDMMC Timings

The Secure Digital Multimedia Card (SDMMC) Controller supports the embedded MultiMedia Card (e.MMC) Specification V4.51, the SD Memory Card Specification V3.0, and the SDIO V3.0 specification. It is compliant with the SD Host Controller Standard V3.0 specification.

Features are different for the two instances of SDMMC:

SDMMC0: SD 3.0, eMMC 4.51, 8 bits

SDMMC1: SD 2.0, eMMC 4.41, 4 bits only

In SDR104 mode (SD 3.0), SDMMC0 is limited to 120 MHz (instead of 208 MHz). In HS200 mode (eMMC 4.51), SDMMC0 is limited to 120 MHz (instead of 200 MHz).

## 66.26 GMAC Timings

### 66.26.1 Timing Conditions

Timings assuming a capacitance load on data and clock are given in the table below.

**Table 66-95. Capacitance Load on Data, Clock Pads**

| Supply | Corner | |
|---|---|---|
| | Max | Min |
| 3.3V | 20 pF | 0 pF |

### 66.26.2 Timing Constraints

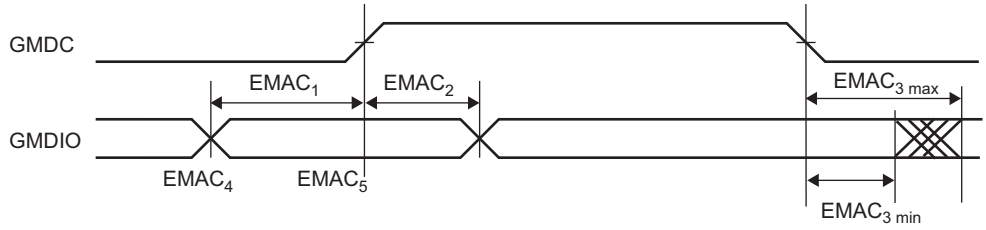
**Table 66-96. Ethernet MAC Signals Relative to GMDC**

| Symbol | Parameter | Min | Max | Unit |
|---|---|---|---|---|
| $EMAC_1$ | Setup for GMDIO from GMDC rising | 10 | – | ns |
| $EMAC_2$ | Hold for GMDIO from GMDC rising | 10 | – | ns |
| $EMAC_3$ | GMDIO toggling from GMDC rising(1) | 0 | 300 | ns |

**Note:**

1. For Ethernet MAC output signals, minimum and maximum access times are defined. The minimum access time is the time between the GMDC rising edge and the signal change. The maximum access time is the time between the GMDC rising edge and the signal stabilizes. The figure below illustrates minimum and maximum accesses for $EMAC_3$.

**Figure 66-40. Minimum and Maximum Access Times of Ethernet MAC Output Signals**



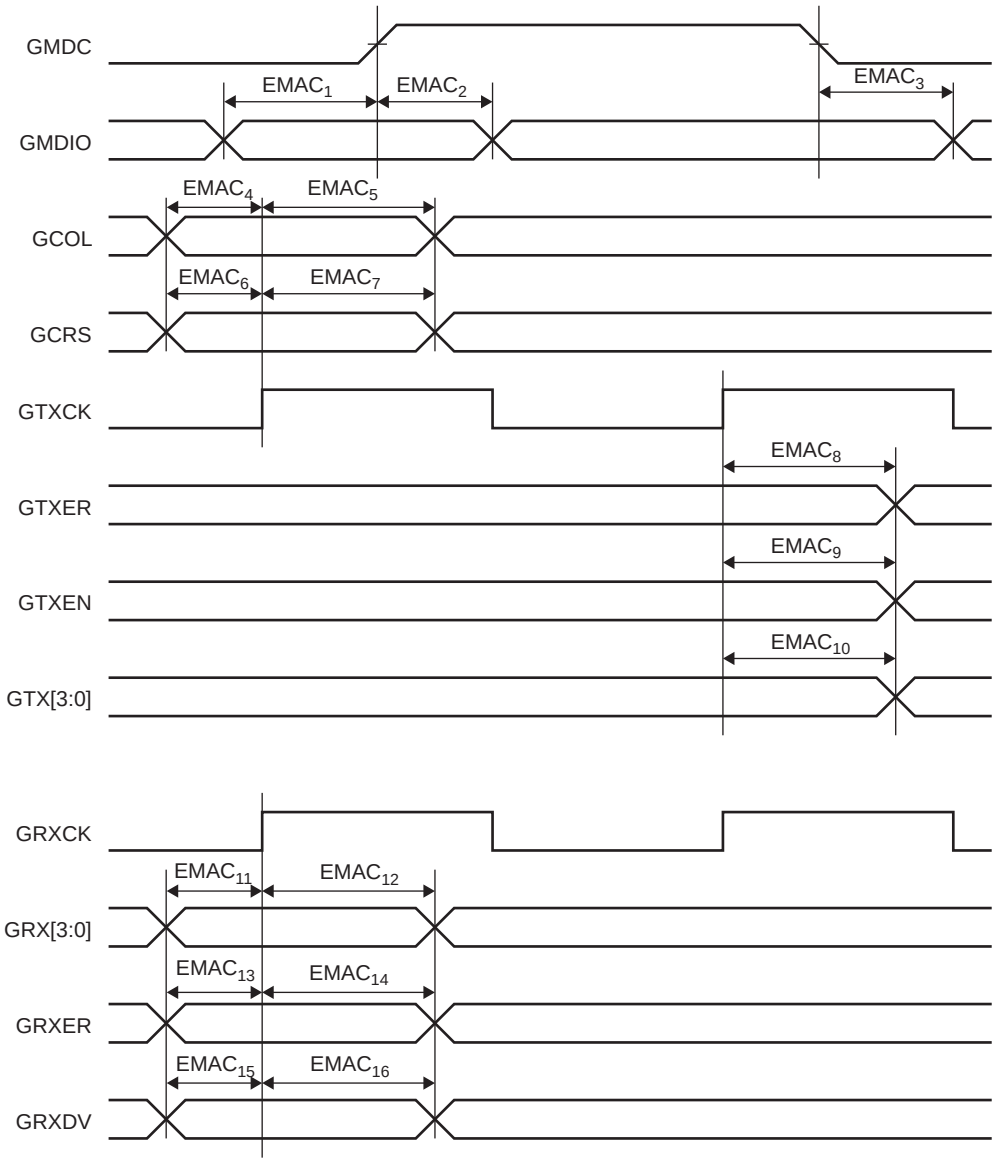

#### 66.26.2.1 Ethernet MAC MII Mode

**Table 66-97. Ethernet MAC MII Specific Signals**

| Symbol | Parameter | Min | Max | Unit |
|---|---|---|---|---|
| $EMAC_4$ | Setup for GCOL from GTXCK rising | 10 | – | ns |
| $EMAC_5$ | Hold for GCOL from GTXCK rising | 10 | – | ns |
| $EMAC_6$ | Setup for GCRS from GTXCK rising | 10 | – | ns |
| $EMAC_7$ | Hold for GCRS from GTXCK rising | 10 | – | ns |
| $EMAC_8$ | GTXER toggling from GTXCK rising | 10 | 25 | ns |
| $EMAC_9$ | GTXEN toggling from GTXCK rising | 10 | 25 | ns |

| .....continued | | | | |
|---|---|---|---|---|
| **Symbol** | **Parameter** | **Min** | **Max** | **Unit** |
| $EMAC_{10}$ | GTX toggling from GTXCK rising | 10 | 25 | ns |
| $EMAC_{11}$ | Setup for GRX from GRXCK | 10 | – | ns |
| $EMAC_{12}$ | Hold for GRX from GRXCK | 10 | – | ns |
| $EMAC_{13}$ | Setup for GRXER from GRXCK | 10 | – | ns |
| $EMAC_{14}$ | Hold for GRXER from GRXCK | 10 | – | ns |
| $EMAC_{15}$ | Setup for GRXDV from GRXCK | 10 | – | ns |
| $EMAC_{16}$ | Hold for GRXDV from GRXCK | 10 | – | ns |

**Figure 66-41. Ethernet MAC MII Mode**

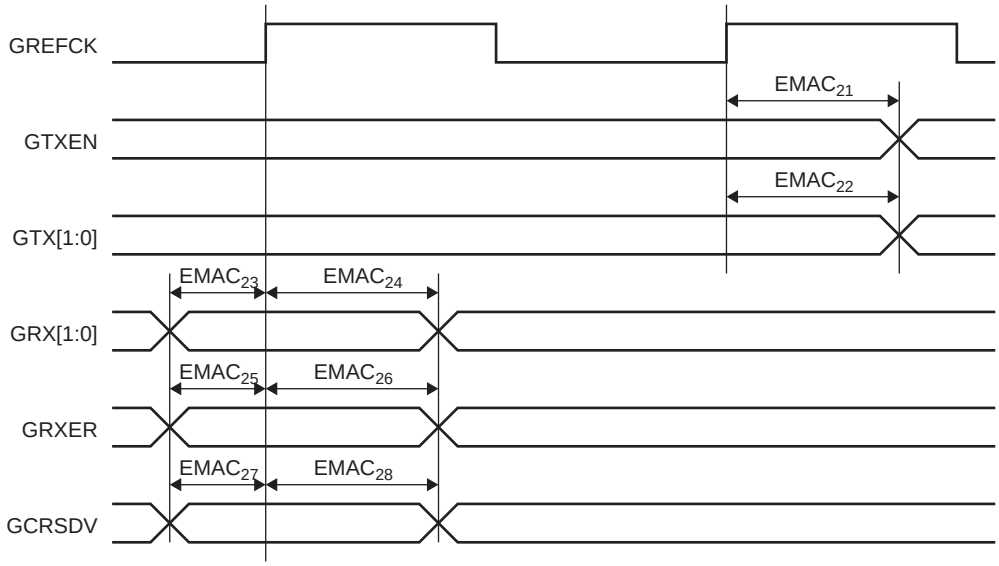


#### 66.26.2.2 Ethernet MAC RMII Mode

**Table 66-98. Ethernet MAC RMII Mode**

| Symbol | Parameter | Min | Max | Unit |
|---|---|---|---|---|
| $EMAC_{21}$ | GTXEN toggling from GREFCK rising | 2 | 16 | ns |
| $EMAC_{22}$ | GTX toggling from GREFCK rising | 2 | 16 | ns |
| $EMAC_{23}$ | Setup for GRX from GREFCK rising | 4 | – | ns |
| $EMAC_{24}$ | Hold for GRX from GREFCK rising | 2 | – | ns |
| $EMAC_{25}$ | Setup for GRXER from GREFCK rising | 4 | – | ns |
| $EMAC_{26}$ | Hold for GRXER from GREFCK rising | 2 | – | ns |
| $EMAC_{27}$ | Setup for GCRSDV from GREFCK rising | 4 | – | ns |
| $EMAC_{28}$ | Hold for GCRSDV from GREFCK rising | 2 | – | ns |

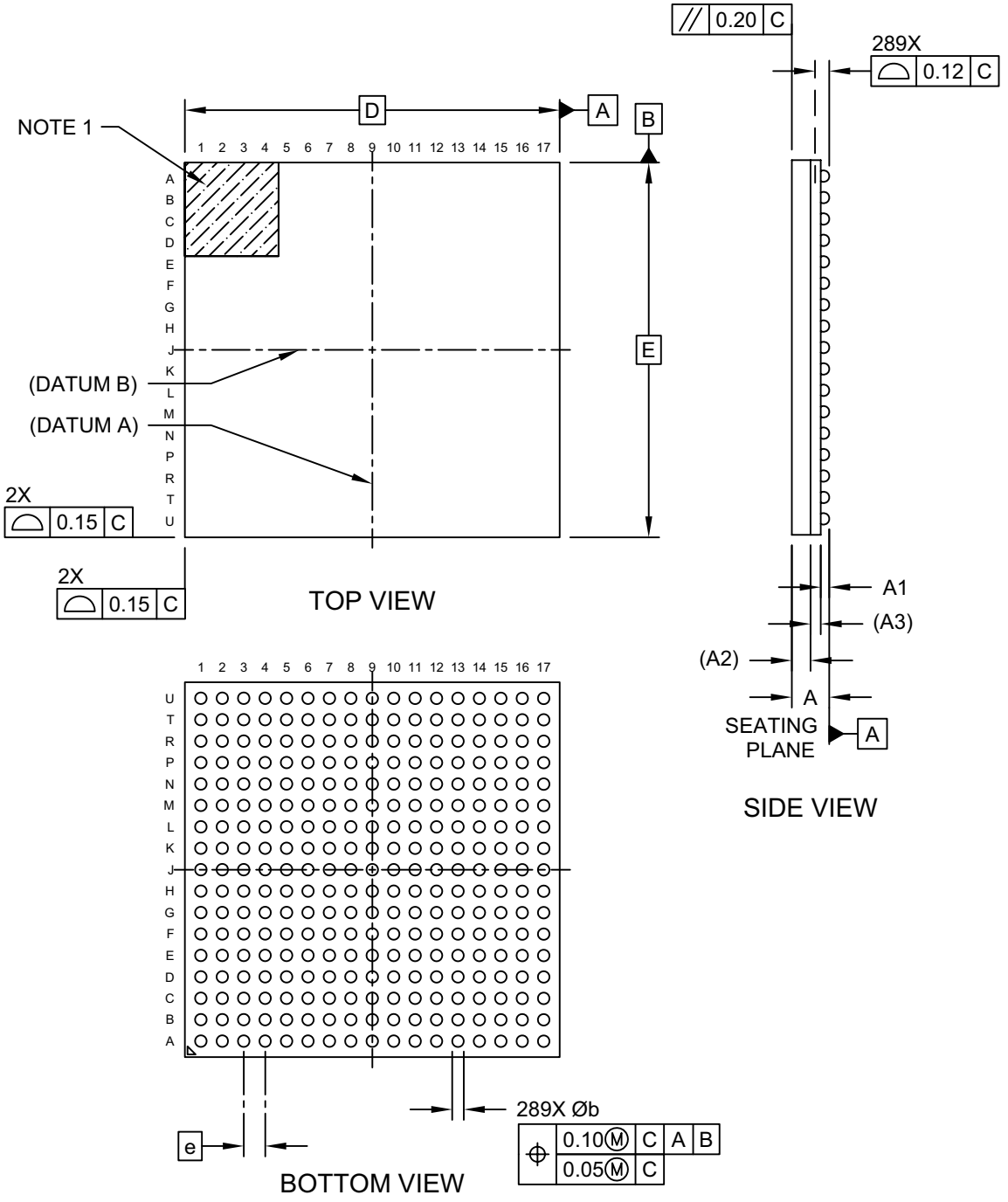
**Figure 66-42. Ethernet MAC RMII Timings**



# 67. Mechanical Characteristics

## 67.1 289-Ball Low Profile Fine Pitch Ball Grid Array (AMB) - 14x14x1.4 mm Body [LFBGA] Atmel Legacy Global Package Code CCZ

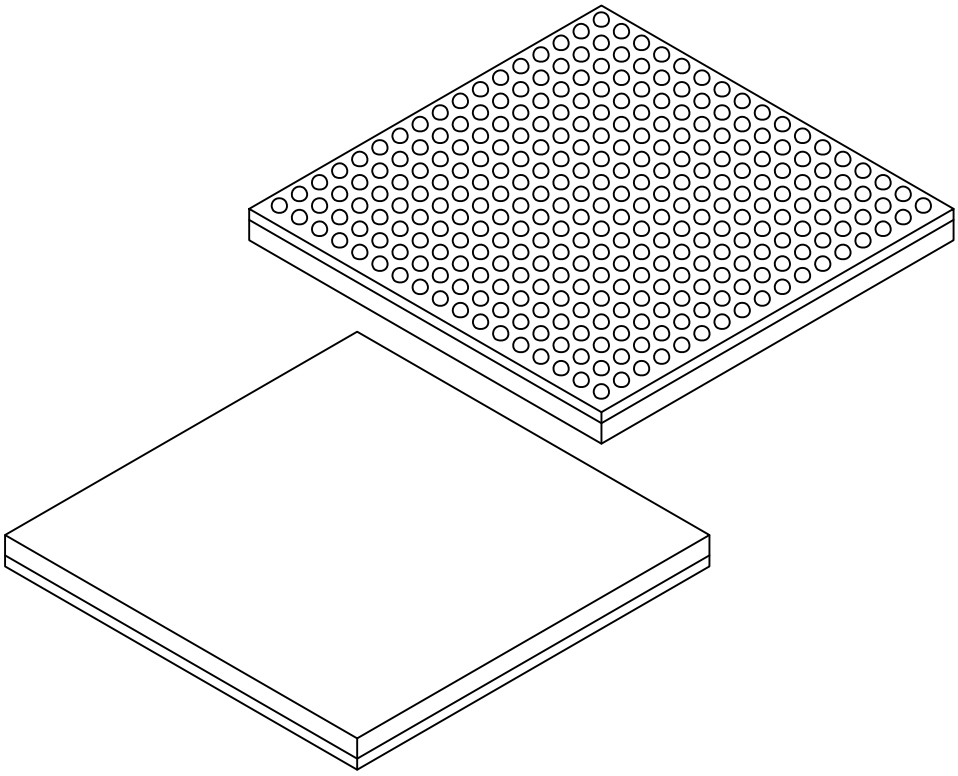
**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




Microchip Technology Drawing C04-21124 Rev A Sheet 1 of 2

## 289-Ball Low Profile Fine Pitch Ball Grid Array (AMB) - 14x14x1.4 mm Body [LFBGA] Atmel Legacy Global Package Code CCZ

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Terminals | N | | 289 | |
| Pitch | e | | 0.80 BSC | |
| Overall Height | A | - | - | 1.40 |
| Ball Height | A1 | 0.27 | 0.32 | 0.37 |
| Molded Cap Thickness | A2 | | 0.70 REF | |
| Substrate Thickness | A3 | | 0.26 REF | |
| Overall Length | D | | 14.00 BSC | |
| Overall Width | E | | 14.00 BSC | |
| Terminal Diameter | b | 0.38 | 0.43 | 0.48 |

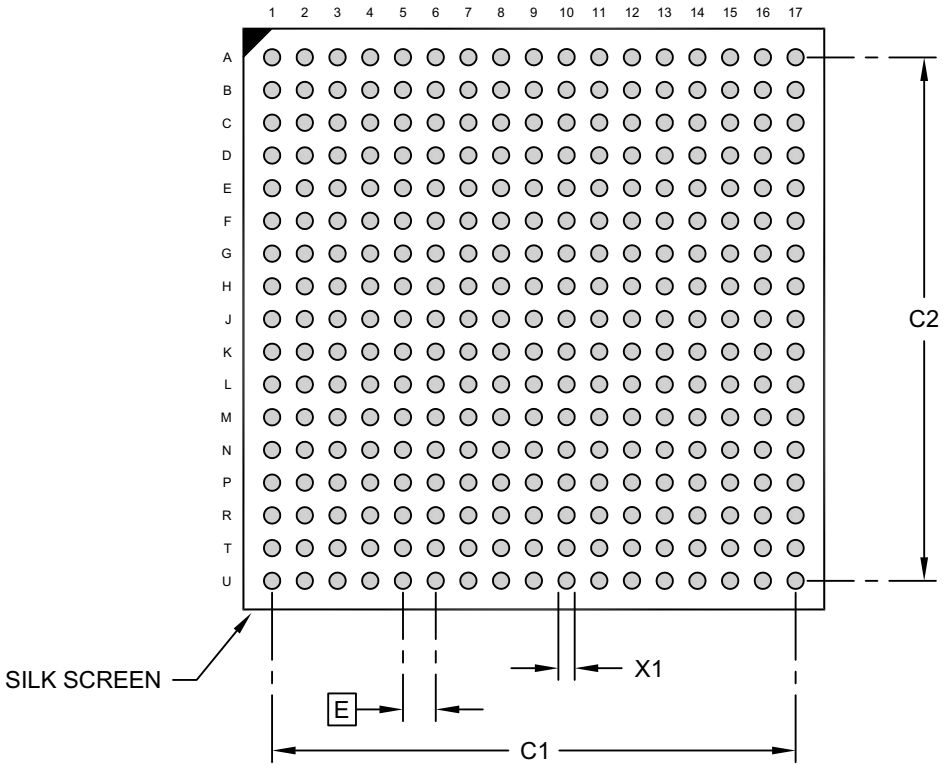
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-21124 Rev A Sheet 1 of 2

### 289-Ball Low Profile Fine Pitch Ball Grid Array (AMB) - 14x14x1.4 mm Body [LFBGA] Atmel Legacy Global Package Code CCZ

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




RECOMMENDED LAND PATTERN

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | 0.80 BSC | | |
| Contact Pad Spacing | C1 | | 12.80 | |
| Contact Pad Spacing | C2 | | 12.80 | |
| Contact Pad Width (X20) | X1 | | | 0.40 |

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
2. For best soldering results, thermal vias, if used, should be filled or tented to avoid solder loss during reflow process

Microchip Technology Drawing C04-23124 Rev A

**Table 67-1. 289-ball LFBGA Package Characteristics**

| | |
|---|---|
| Moisture Sensitivity Level | 3 |

**Table 67-2. Device and 289-ball LFBGA Package Weight**

| | |
|---|---|
| 445 | mg |

**Table 67-3. Package Reference**

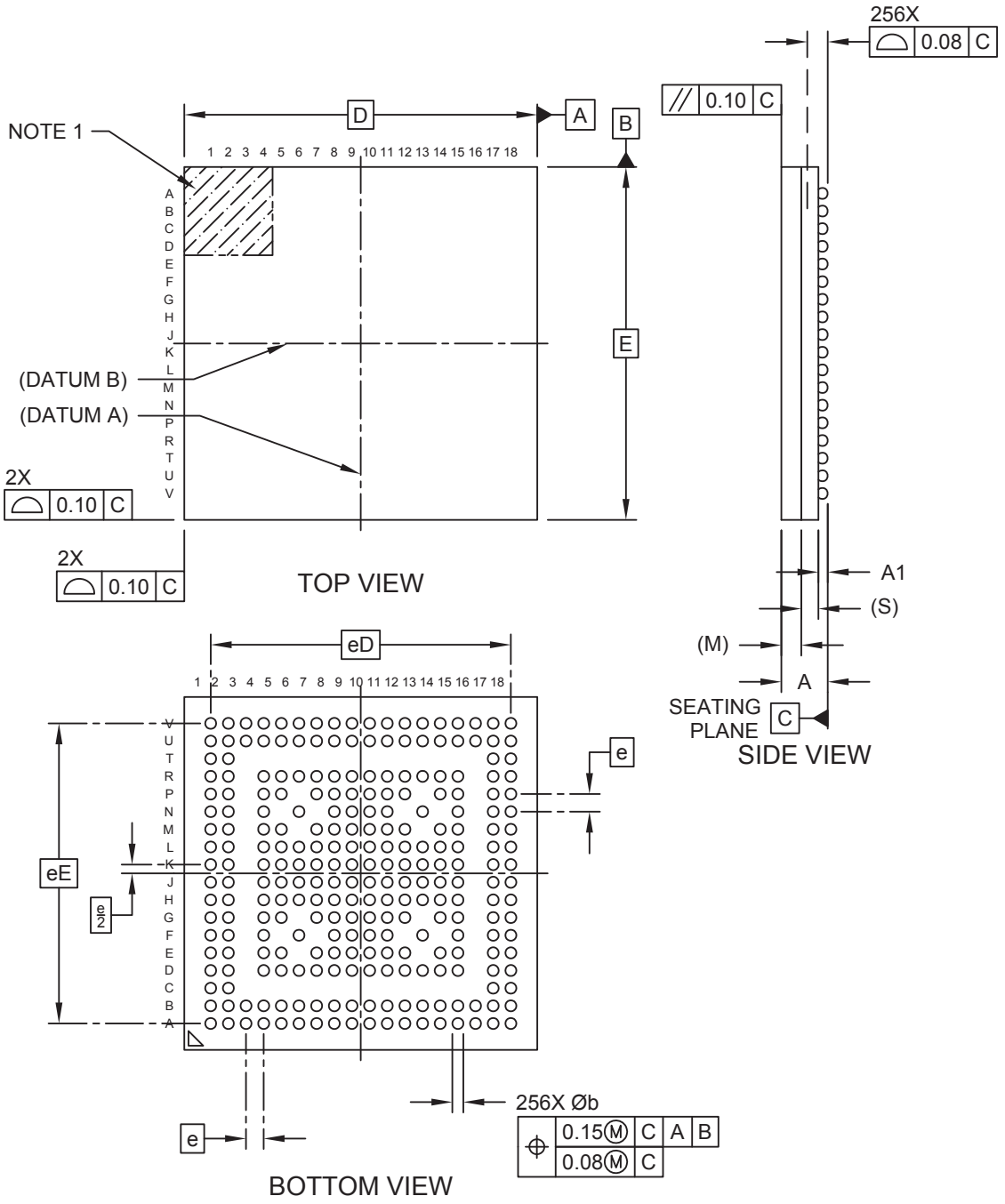
| | |
|---|---|
| JEDEC Drawing Reference | NA |
| J-STD-609 Classification | e8 |

**Table 67-4. 289-ball LFBGA Package Information**

| | |
|---|---|
| Ball Land | 0.450 mm ±0.05 |
| Nominal Ball Diameter | 0.4 mm |
| Solder Mask Opening | 0.350 mm ±0.05 |
| Solder Mask Definition | SMD |
| Solder | OSP |

## 67.2 256-Ball Thin Fine Pitch Ball Grid Array (AYB) - 8x8x1.05 mm Body [TFBGA]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging

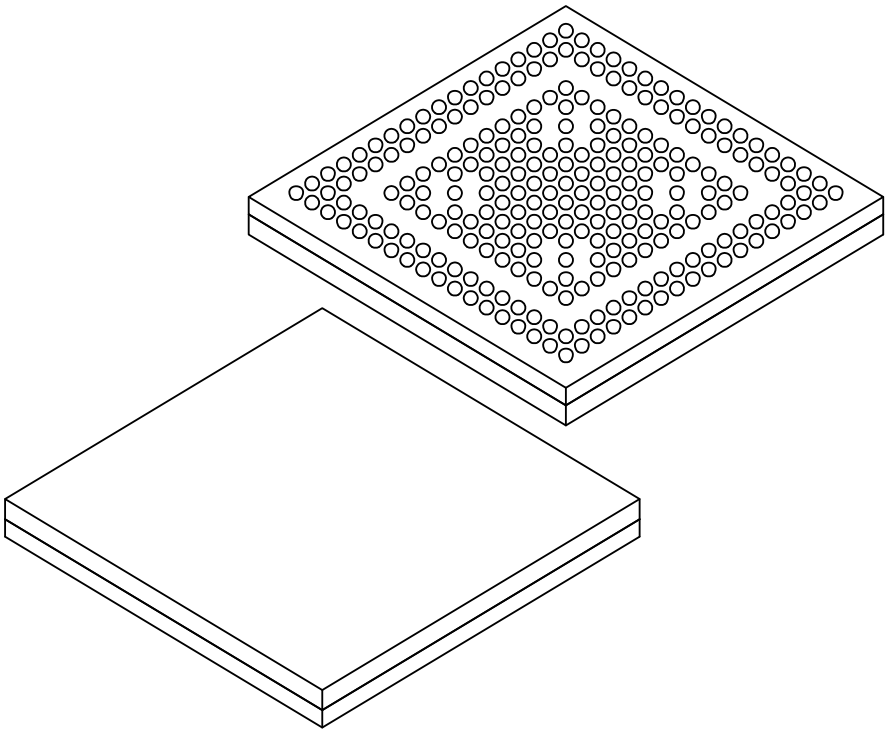



Microchip Technology Drawing C04-21133 Rev A Sheet 1 of 2

© 2018 Microchip Technology Inc.

## 256-Ball Thin Fine Pitch Ball Grid Array (AYB) - 8x8x1.05 mm Body [TFBGA]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Terminals | N | | 256 | |
| Pitch | e | | 0.40 BSC | |
| Overall Height | A | - | - | 1.05 |
| Ball Height | A1 | 0.11 | 0.16 | 0.21 |
| Substrate Thickness | S | | 0.26 REF | |
| Mold Cap Thickness | M | | 0.45 REF | |
| Overall Length | D | | 8.00 BSC | |
| Overall Pitch | eD | | 6.80 BSC | |
| Overall Width | E | | 8.00 BSC | |
| Overall Pitch | eE | | 6.80 BSC | |
| Ball Diameter | b | 0.20 | 0.25 | 0.30 |

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensioning and tolerancing per ASME Y14.5M

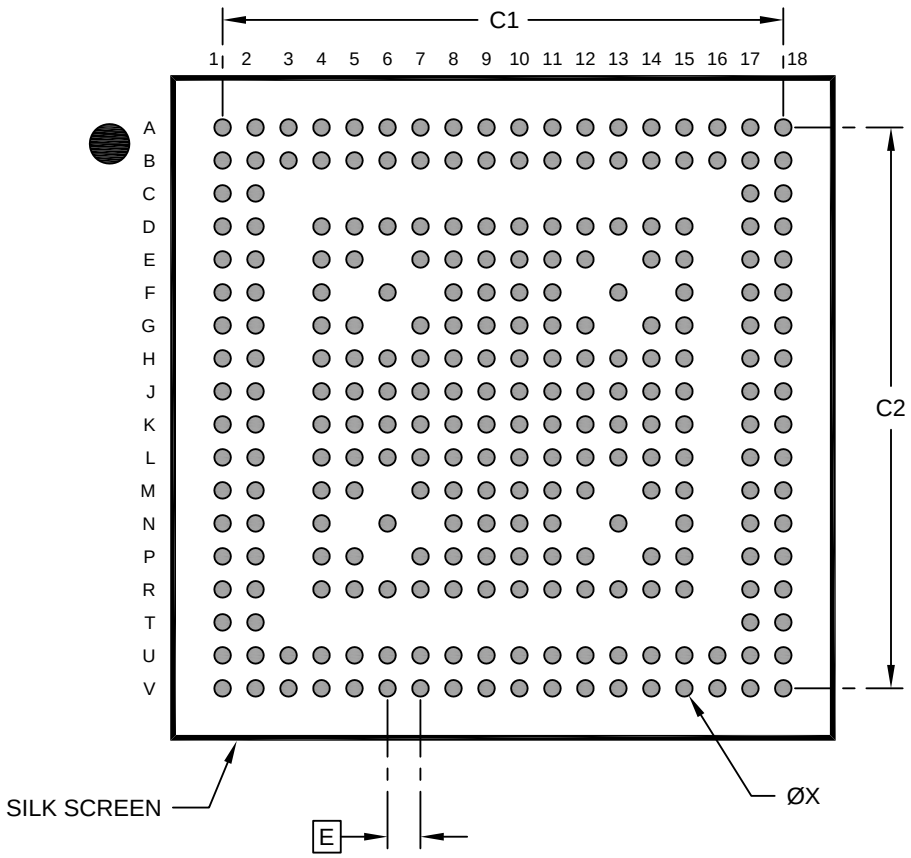
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-21133 Rev A Sheet 2 of 2

© 2018 Microchip Technology Inc.

### 256-Ball Thin Fine Pitch Ball Grid Array (AYB) - 8x8x1.05 mm Body [TFBGA]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




RECOMMENDED LAND PATTERN

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | 0.40 BSC | | |
| Contact Pad Spacing | C1 | | 6.80 | |
| Contact Pad Spacing | C2 | | 6.80 | |
| Contact Pad Diameter (X256) | X | | | 0.20 |

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-23133 Rev A

© 2018 Microchip Technology Inc.

**Table 67-5. 256-ball TFBGA Package Characteristics**

| | |
|---|---|
| Moisture Sensitivity Level | 3 |

**Table 67-6. Device and 256-ball TFBGA Package Weight**

| | |
|---|---|
| 110.3 | mg |

**Table 67-7. Package Reference**

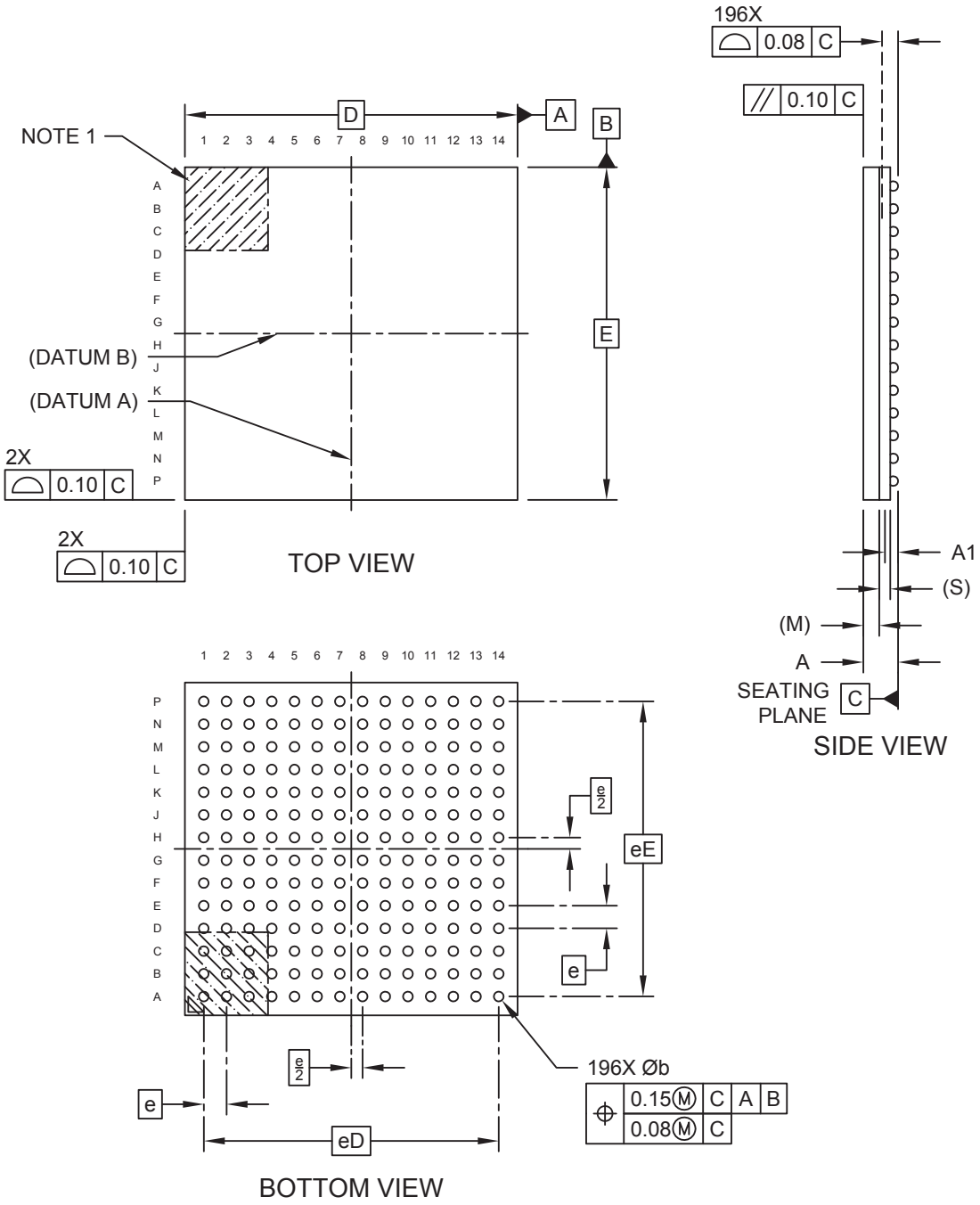
| | |
|---|---|
| JEDEC Drawing Reference | NA |
| J-STD-609 Classification | e8 |

**Table 67-8. 256-ball TFBGA Package Information**

| | |
|---|---|
| Ball Land | 0.350 mm ±0.05 |
| Nominal Ball Diameter | 0.25 mm |
| Solder Mask Opening | 0.250 mm ±0.05 |
| Solder Mask Definition | SMD |
| Solder | OSP |

## 67.3 196-Ball Thin Fine Pitch Ball Grid Array (BAB) - 11x11 mm Body [TFBGA]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging

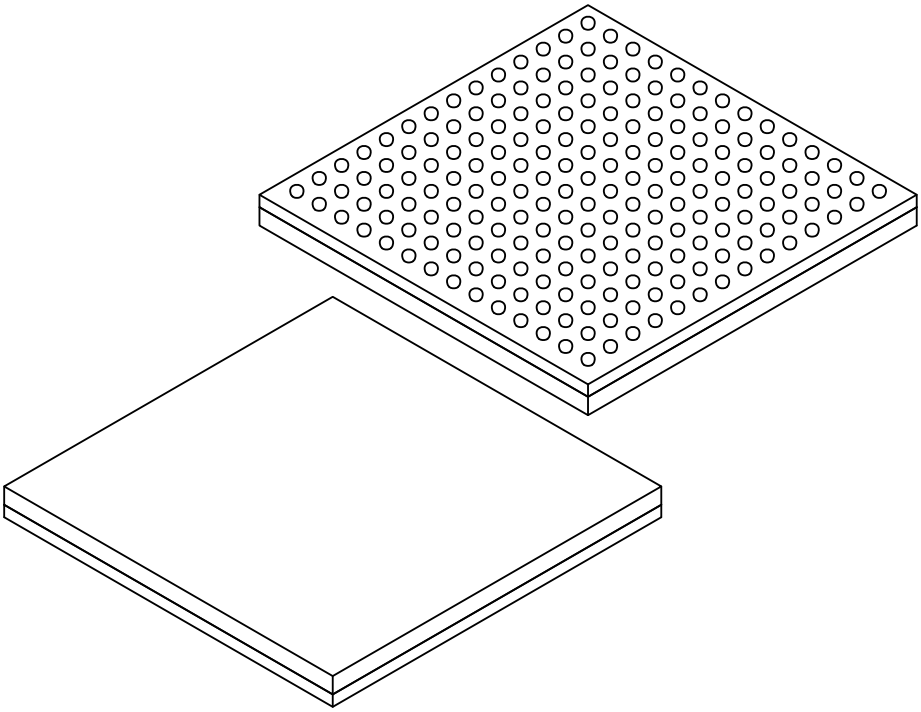



Microchip Technology Drawing C04-21141 Rev A Sheet 1 of 2

© 2018 Microchip Technology Inc.

## 196-Ball Thin Fine Pitch Ball Grid Array (BAB) - 11x11 mm Body [TFBGA]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Terminals | N | | 196 | |
| Pitch | e | | 0.75 BSC | |
| Overall Height | A | - | - | 1.20 |
| Ball Height | A1 | 0.16 | 0.25 | 0.26 |
| Substrate Thickness | S | | 0.36 REF | |
| Mold Cap Thickness | M | | 0.53 REF | |
| Overall Length | D | | 11.00 BSC | |
| Overall Ball Pitch | eD | | 9.75 BSC | |
| Overall Width | E | | 11.00 BSC | |
| Overall Ball Pitch | eE | | 9.75 BSC | |
| Ball Diameter | b | 0.27 | 0.32 | 0.37 |

Notes:

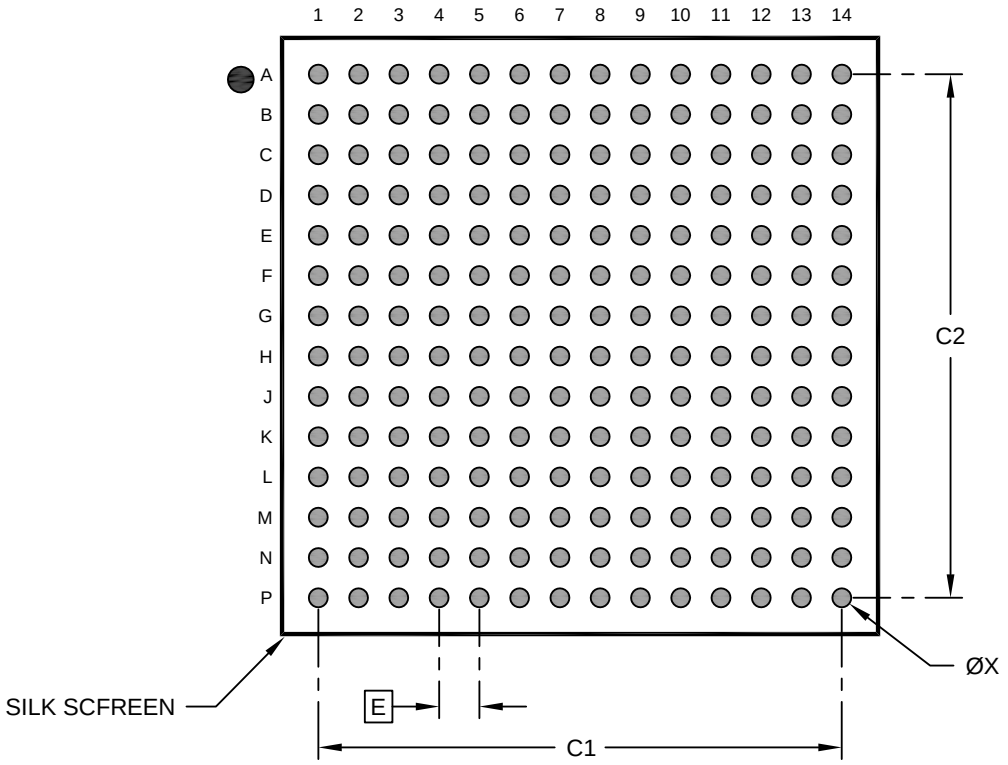
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-21141 Rev A Sheet 2 of 2

© 2018 Microchip Technology Inc.

### 196-Ball Thin Fine Pitch Ball Grid Array (BAB) - 11x11 mm Body [TFBGA]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




RECOMMENDED LAND PATTERN

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | 0.75 BSC | | |
| Contact Pad Spacing | C1 | | 9.75 | |
| Contact Pad Spacing | C2 | | 9.75 | |
| Contact Pad Diameter (X196) | X | | | 0.35 |

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-23141 Rev A

© 2018 Microchip Technology Inc.

**Table 67-9. 196-ball TFBGA Package Characteristics**

| | |
|---|---|
| Moisture Sensitivity Level | 3 |

**Table 67-10. Device and 196-ball TFBGA Package Weight**

| | |
|---|---|
| 234.2 | mg |

**Table 67-11. Package Reference**

| | |
|---|---|
| JEDEC Drawing Reference | NA |
| J-STD-609 Classification | e8 |

**Table 67-12. 196-ball TFBGA Package Information**

| | |
|---|---|
| Ball Land | 0.350 mm ±0.05 |
| Nominal Ball Diameter | 0.3 mm |
| Solder Mask Opening | 0.275 mm ±0.30 |
| Solder Mask Definition | SMD |
| Solder | OSP |

# 68. Schematic Checklist

The schematic checklist provides the user with the requirements regarding the different pin connections that must be considered before starting any new board design as well as information on the minimum hardware resources required to quickly develop an application with the SAMA5D2. It does not consider PCB layout constraints.

It also provides recommendations regarding low-power design constraints to minimize power consumption.

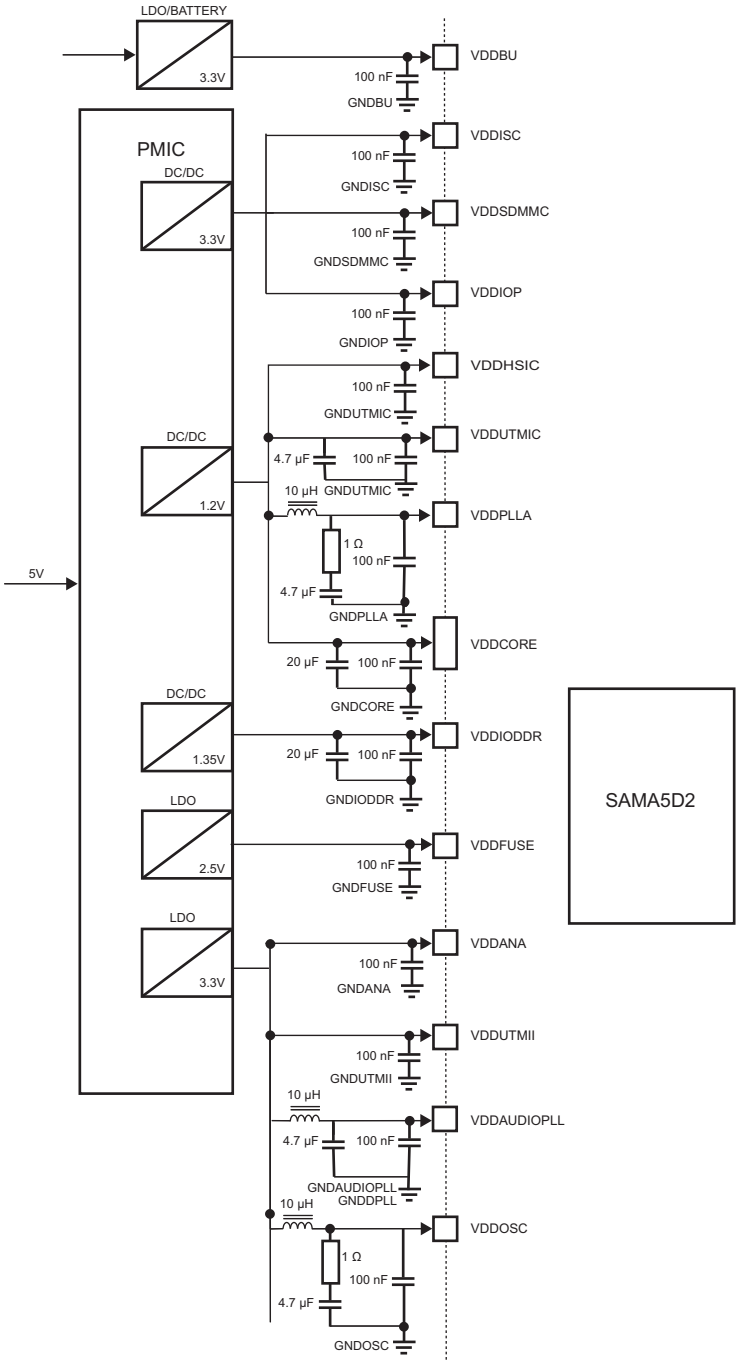
This information is not intended to be exhaustive. Its objective is to cover as many configurations of use as possible.

The checklist contains a column for use by designers, making it easy to track and verify each line item.

## 68.1 Power Supply

**⚠ CAUTION** The board design must comply with the power-up and power-down sequence guidelines provided in the data sheet to ensure reliable operation of the device.

**Figure 68-1. 1.2V, 1.35V/1.5V, 2V, 2.5V, 3.3V Power Supplies Schematics(1)**




**Note:**

1. These values are given only as typical examples.

**Table 68-1. Power Supply Connections**

| ☑ | Signal Name | Recommended Pin Connection | Description |
|---|---|---|---|
| | VDDCORE | 1.1V to 1.32V<br>Decoupling/Filtering capacitors (10 µF and 100 nF)(1)(2) | Powers the core<br>Supply ripple must not exceed 15 mVrms. |
| | VDDPLLA | 1.1V to 1.32V<br>Decoupling/Filtering capacitor (100 nF)(1)(2) | Powers the PLLA cell.<br>The VDDPLLA power supply pin draws small current, but it is noise-sensitive. Care must be taken in VDDPLLA power supply routing, decoupling and also on bypass capacitors.<br>Supply ripple must not exceed 20 mVrms. |
| | VDDIODDR | 1.14V to 1.30V<br>1.283V to 1.45V<br>1.425V to 1.575V<br>1.7V to 1.9V<br>Decoupling/Filtering capacitors (10 µF and 100 nF)(1)(2) | Powers LPDDR2-LPDDR3 interface<br>Powers DDR3L interface<br>Powers DDR3 interface<br>Powers LPDDR1-DDR2 interface<br>Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop. |
| | VDDISC | 1.65V to 3.6V<br>Decoupling capacitor (100 nF) (1)(2) | Powers the ISC Interface I/O lines. |
| | VDDIOP0,1,2 | 1.65V to 3.6V<br>Decoupling capacitor (100 nF) (1)(2) | Powers the peripherals I/O lines.<br>Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop. |
| | VDDBU | 1.65V to 3.6V<br>Decoupling capacitor (100 nF) (1)(2) | Powers the Slow Clock oscillator, the internal 64 kHz RC and a part of the System Controller.<br>Must be established first.<br>Supply ripple must not exceed 30 mVrms. |
| | VDDUTMIC | 1.1V to 1.32V<br>Decoupling capacitor (100 nF) (1)(2) | DC Supply UTMI Phy (Core) and PLL UTMI<br>Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop. |

.....continued

| ☑ | Signal Name | Recommended Pin Connection | Description |
|---|---|---|---|
| | VDDUTMII | 3.0V to 3.6V<br>Decoupling capacitor (100 nF) (1)(2) | DC Supply UTMI Phy (Interface)<br>Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop. |
| | VDDHSIC | 1.1V to 1.3V<br>Decoupling capacitors (100 nF) (1)(2) | DC Supply HSIC Phy |
| | VDDOSC | 1.65V to 3.6V<br>Decoupling/Filtering RLC circuit(1) | Powers the main oscillator cell.<br>The VDDOSC power supply pin is noise-sensitive. Care must be taken in VDDOSC power supply routing, decoupling and also on bypass capacitors.<br>Supply ripple must not exceed 30 mVrms. |
| | VDDSDMMC | 1.65V to 3.6V<br>Decoupling capacitor (100 nF) (1)(2) | Powers the SDMMC I/O lines. |
| | VDDANA | 1.65V to 3.6V<br>Decoupling capacitor (100 nF) (1)(2) | Powers the analog parts. |
| | VDDFUSE | 2.25V to 2.75V<br>Decoupling capacitor (100 nF) (1)(2) | Powers the fuse box for programming.<br>VDDFUSE must not be left floating. |
| | VDDAUDIOPLL | 3.0V to 3.6V<br>Decoupling capacitor (100 nF) (1)(2) | Powers the Audio PLL. |
| | GNDCORE | Core Chip ground | GNDCORE pins are common to VDDCORE pins.<br>GNDCORE pins should be connected as shortly as possible to the system ground plane. |

.....continued

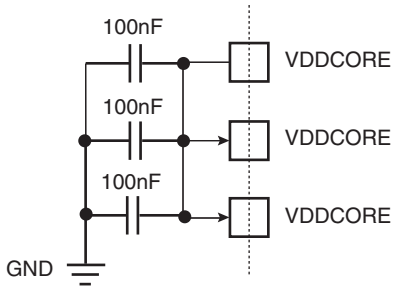
| ☑ | Signal Name | Recommended Pin Connection | Description |
|---|---|---|---|
| | GNDPLLA | PLLA cell ground | GNDPLL pin is provided for VDDPLLA pins.<br>GNDPLL pin should be connected as shortly as possible to the system ground plane. |
| | GNDIODDR | DDR2/LPDDR1/LPDDR2/ DDR3/LPDDR3 interface I/O lines ground | GNDIODDR pins should be connected as shortly as possible to the system ground plane. |
| | GNDISC | VDDISC ground | GNDISC pins are common to VDDISC pins.<br>GNDISC pins should be connected as shortly as possible to the system ground plane. |
| | GNDIOP0,1,2 | Peripherals and ISC I/O lines ground | GNDIOPx pins are common to VDDIOPx pins.<br>GNDIOP pins should be connected as shortly as possible to the system ground plane. |
| | GNDBU | Backup ground | GNDBU pin is provided for VDDBU pins.<br>GNDBU pin should be connected as shortly as possible to the system ground plane. |
| | GNDUTMIC | VDDUTMIC and VDDHSIC ground | GNDUTMIC pins are common to VDDUTMIC and VDDHSIC pins.<br>GNDUTMIC pins should be connected as shortly as possible to the system ground plane. |
| | GNDUTMII | UDPHS and UHPHS UTMI+ Core and Interface, and PLL UTMI ground | GNDUTMII pins are common to VDDUTMII and VDDUTMIC pins.<br>GNDUTMII pins should be connected as shortly as possible to the system ground plane. |
| | GNDOSC | Oscillator ground | GNDOSC pin is provided for VDDOSC pins.<br>GNDOSC pin should be connected as shortly as possible to the system ground plane. |

.....continued

| ☑ | Signal Name | Recommended Pin Connection | Description |
|---|---|---|---|
| | GNDSDMMC | SDMMC ground | SDMMC pins are common to VDDSDMMC pins.<br>GNDSDMMC pins should be connected as shortly as possible to the system ground plane. |
| | GNDANA | Analog ground | GNDANA pins are common to VDDANA pins.<br>GNDANA pins should be connected as shortly as possible to the system ground plane. |
| | GNDFUSE | Fuse box ground | GNDFUSE pins are common to VDDFUSE pins.<br>GNDFUSE pins should be connected as shortly as possible to the system ground plane. |
| | GNDAUDIOPLL<br>GNDDPLL | Audio PLL ground | GNDAUDIOPLL and GNDDPLL pins are common to VDDAUDIOPLL.<br>GNDAUDIOPLL and GNDDPLL pins should be connected as shortly as possible to the system ground plane. |

**Note:**

1. These values are given only as typical examples.
2. Decoupling capacitors must be connected as close as possible to the microprocessor and on each relevant pin



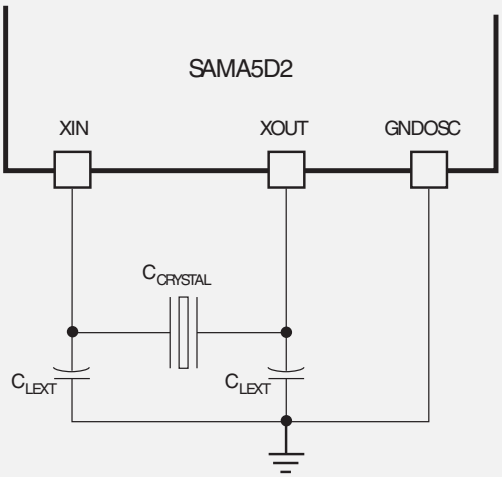

For more information, see the table “VDDCORE Power-On Reset Characteristics” in the section "Electrical Characteristics".

## 68.2 Power-On Reset

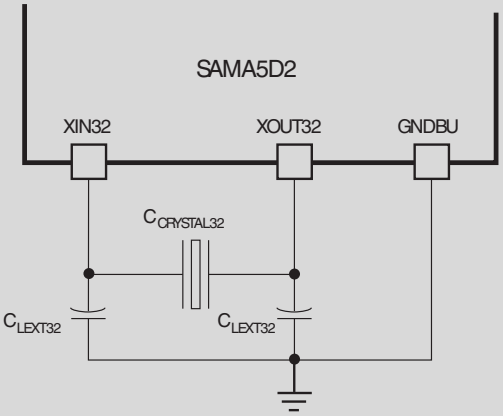
The SAMA5D2 embeds several Power-On Resets (PORs) to ensure the power supply is established when the reset is released. These PORs are dedicated to VDDBU, VDDANA and VDDCORE respectively.

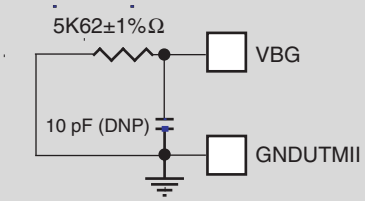
## 68.3 Clock, Oscillator and PLL

**Table 68-2. Clock, Oscillator and PLL Connections**

| ☑ | Signal Name | Recommended Pin Connection | Description |
|---|---|---|---|
| | XIN<br>XOUT<br>Main Oscillator in Normal Mode | Crystals between 8 and 24 MHz<br>USB High Speed (not Full Speed) Host and Device peripherals need a 12 MHz clock.<br>Capacitors on XIN and XOUT (Crystal Load Capacitance dependent) | Crystal Load Capacitance to check ($C_{CRYSTAL}$)<br>  <br>Refer to the section Electrical Characteristics. |
| | XIN<br>XOUT<br>Main Oscillator in Bypass Mode | XIN: external clock source<br>XOUT: can be left unconnected<br>USB High speed (not Full Speed) Host and Device peripherals need a 12 MHz clock. | Square wave signal, high level = VDDOSC<br>External clock source up to 50 MHz<br>Duty Cycle: 40 to 60%<br>Refer to the section Electrical Characteristics. |

.....continued

| ☑ | Signal Name | Recommended Pin Connection | Description |
|---|---|---|---|
| | XIN<br>XOUT<br>Main Oscillator Disabled | XIN: can be left unconnected<br>XOUT: can be left unconnected<br>USB High Speed (not Full Speed)<br>Host and Device peripherals need a 12 MHz clock. | Typical nominal frequency 12 MHz (Internal 12 MHz RC Oscillator)<br>Duty Cycle: 45 to 55%<br>Refer to the section Electrical Characteristics. |
| | XIN32<br>XOUT32<br>Slow Clock Oscillator | 32.768 kHz Crystal<br>Capacitors on XIN32 and XOUT32<br>(Crystal Load Capacitance dependent) | Crystal load capacitance to check ($C_{CRYSTAL32}$)<br>  <br>Refer to the section Electrical Characteristics. |
| | XIN32<br>XOUT32<br>Slow Clock Oscillator Disabled | XIN32: can be left unconnected<br>XOUT32: can be left unconnected | Typical nominal frequency 32 kHz (internal 64 kHz RC oscillator)<br>Duty Cycle: 45 to 55%<br>Refer to the section Electrical Characteristics. |

| .....continued | | | |
|---|---|---|---|
| ☑ | Signal Name | Recommended Pin Connection | Description |
| | VBG | 0.9–1.1V(2) | Bias Voltage Reference for USB<br>To reduce as much as possible the noise on VBG pin, check the layout considerations below:<br>- VBG path as short as possible<br>- Ground connection to GNDUTMII<br>  <br>VBG can be left unconnected if USB is not used.<br>Refer to the section Signal Description. |

## 68.4 ICE and JTAG

**Note:** It is recommended to establish accessibility to a JTAG connector for debug in any case.

**Note:** The boundary scan is only available on JTAG IO Set 1 signals.

**Table 68-3. ICE and JTAG Connections**

| ☑ | Signal Name | Recommended Pin Connection | Description |
|---|---|---|---|
| | TCK | Pull-up (100 kΩ)(1)<br>If Debug mode is not required, this pin can be used as GPIO. | This pin is a Schmitt trigger input.<br>Internal pull-up resistor to $V_{DDIOP}$ (100 kΩ). |
| | TMS | Pull-up (100 kΩ)(1)<br>If Debug mode is not required, this pin can be used as GPIO. | This pin is a Schmitt trigger input.<br>Internal pull-up resistor to $V_{DDIOP}$ (100 kΩ). |
| | TDI | Pull-up (100 kΩ)(1)<br>If Debug mode is not required, this pin can be used as GPIO. | This pin is a Schmitt trigger input.<br>Internal pull-up resistor to $V_{DDIOP}$ (100 kΩ). |

.....continued

| ☑ | Signal Name | Recommended Pin Connection | Description |
|---|---|---|---|
| | TDO | Floating<br>If Debug mode is not required, this pin can be used as GPIO. | Output driven at up to $V_{DDIOP}$. |
| | NTRST | Refer to the section Pin Description.<br>If Debug mode is not required, this pin can be used as GPIO. | This pin is a Schmitt trigger input.<br>Internal pull-up resistor to $V_{DDIOP}$ (100 kΩ). |
| | JTAGSEL | In harsh environments(2), it is strongly recommended to tie this pin to GNDBU if not used or to add an external low-value resistor (such as 1 kΩ). | Internal pull-down resistor to GNDBU (15 kΩ).<br>Must be tied to $V_{DDBU}$ to enter JTAG Boundary Scan. |

**Note:**

1. These values are given only as typical examples.
2. In a well-shielded environment subject to low magnetic and electric field interference, the pin may be left unconnected. In noisy environments, a connection to ground is recommended.

## 68.5 Reset and Test

**Table 68-4. Reset and Test Connections**

| ☑ | Signal Name | Recommended Pin Connection | Description |
|---|---|---|---|
| | NRST | Application-dependent<br>Can be connected to a push button for hardware reset.<br>In applications with a storage element on VDDBU (battery, supercapacitor, etc.), NRST must be tied to an non-permanent supply (satisfying VIH conditions on NRST) and not to VDDBU. Typically, this may be the application's primary 3.3V supply (VDDIOP0, VDDIOP1 or any other). The recommended pull-up value is 10 kOhms minimum. Doing so, the NRST level is 'L' when the non-permanent supply is removed. On the contrary, if this pin is tied to VDDBU:<br>• the circuit driving this pin low may not be operational when the input power of the application is removed. Therefore the processor may not be properly reset during power-up or power-down phases,<br>• a leakage path from VDDBU to the application's input power may be created when this input power is removed. (This may be for example through a clamping diode that may be present on an external integrated circuit driving NRST.) | NRST pin is a Schmitt trigger input.<br>No internal pull-up resistor. |
| | TST | In harsh environments(1), it is strongly recommended to tie this pin to GNDBU to add an external low-value resistor (such as 10 kΩ). | This pin is a Schmitt trigger input.<br>Internal pull-down resistor to GNDBU (15 kΩ). |

**Note:**

1. In a well-shielded environment subject to low magnetic and electric field interference, the pin may be left unconnected. In noisy environments, a connection to ground is recommended.

## 68.6 Shutdown/Wake-up Logic

**Table 68-5. Shutdown/Wake-up Logic Connections**

| ☑ | Signal Name | Recommended Pin Connection | Description |
|---|---|---|---|
| | SHDN | Application-dependent<br>A typical application connects pin SHDN to the shutdown input of the DC/DC Converter providing the main power supplies. | Push-pull output. |
| | WKUP | 0V to $V_{DDBU}$ | Input only. WKUP behavior can be configured through the Shutdown Controller (SHDWC). |

## 68.7 Parallel Input/Output (PIO)

**Table 68-6. PIO Connections**

| ☑ | Signal Name | Recommended Pin Connection | Description |
|---|---|---|---|
| | PAx<br>PBx<br>PCx<br>PDx | Application-dependent | All PIOs are pulled up inputs (100 kW) at reset except those which are multiplexed with the Address Bus signals that require to be enabled as peripherals:<br>In the section Package and Pinout, refer to the column ‘Reset State’ of the Pin Description table.<br>Schmitt trigger on all inputs.<br>To reduce power consumption if not used, the relevant PIO can be configured as an output, driven at ‘0’ with internal pull-up disabled. |

## 68.8 Analog-to-Digital Converter (ADC)

**Table 68-7. ADC Connections**

| ☑ | Signal Name | Recommended Pin Connection | Description |
|---|---|---|---|
| | ADVREF | 3.3 V to VDDANA<br>Decoupling/filtering capacitors<br>Application-dependent | ADVREF is a pure analog input.<br>To reduce power consumption if the ADC is not used, connect ADVREF to GNDANA. |

## 68.9 External Bus Interface (EBI)

**Table 68-8. EBI Connections**

| ☑ | Signal Name | Recommended Pin Connection | Description |
|---|---|---|---|
| | D0–D15 | Application-dependent | Data Bus (D0 to D15)<br>All data lines are pulled up inputs at reset. |
| | A0–A25 | Application-dependent | Address Bus (A0 to A25)<br>All address lines are pulled up inputs at reset. |

The tables below detail the connections to be applied between the EBI pins and the external devices for each memory controller.

**Table 68-9. EBI Pins and External Static Devices Connections**

| Signals: EBI_ | Pins of the Interfaced Device | | |
|---|---|---|---|
| | 8-bit Static Devices | 2 x 8-bit Static Devices | 16-bit Static Devices |
| Controller | SMC (Static Memory Controller) | | |
| D0–D7 | D0–D7 | D0–D7 | D0–D7 |
| D8–D15 | – | D8–D15 | D8–D15 |
| A0/NBS0 | A0 | – | NLB |
| A1 | A1 | A0 | A0 |
| A2–A22 | A[2:22] | A[1:21] | A[1:21] |
| A23–A25 | A[23:25] | A[22:24] | A[22:24] |
| NCS0 | CS | CS | CS |
| NCS1 | CS | CS | CS |
| NCS2 | CS | CS | CS |
| NCS3/NANDCS | CS | CS | CS |
| NRD/NANDOE | OE | OE | OE |
| NWE/NWR0/NANDWE | WE | WE(1) | WE |
| NWR1/NBS1 | – | WE(1) | NUB |

**Note:**

1. NWR0 enables lower byte writes. NWR1 enables upper byte writes.

**Table 68-10. EBI Pins and NAND Flash Device Connections**

| Signals: EBI_ | Pins of the Interfaced Device | |
|---|---|---|
| | 8-bit NAND Flash | 16-bit NAND Flash |
| Controller | NFC (NAND Flash Controller) | |
| D0–D7 | NFD0–NFD7 | NFD0–NFD7 |
| D8–D15 | – | NFD8–NFD15 |
| A21/NANDALE | ALE | ALE |
| A22/NANDCLE | CLE | CLE |
| NRD/NANDOE | RE | RE |
| NWE/NWR0/NANDWE | WE | WE |
| NCS3/NANDCS | CE | CE |
| NANDRDY | R/B | R/B |
| A0/NBS0 | – | – |
| A1–A20 | – | – |
| A23–A25 | – | – |
| NWR1/NBS1 | – | – |
| NCS0 | – | – |
| NCS1 | – | – |
| NCS2 | – | – |
| NWAIT | – | – |

## 68.10 USB High-Speed Host Port (UHPHS) / USB High-Speed Device Port (UDPHS)

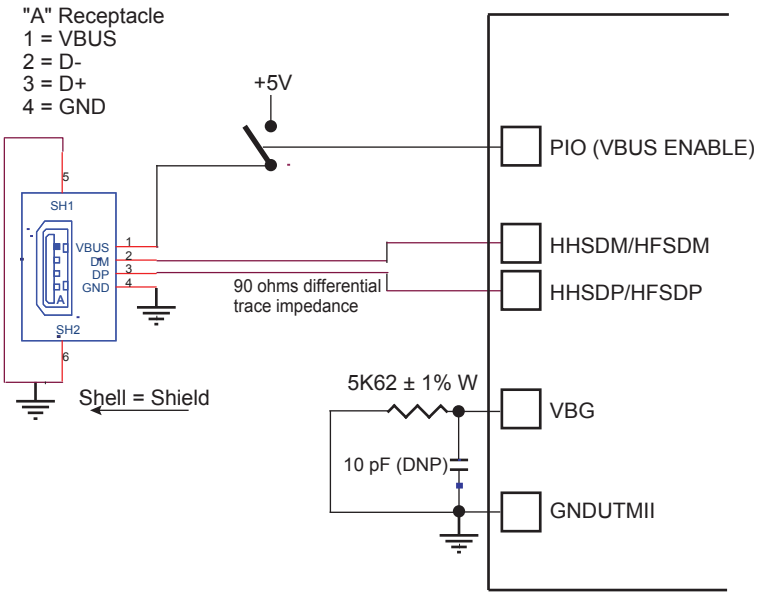
**Table 68-11. UHPHS/UDPHS Connections**

| | Signal Name | Recommended Pin Connection | Description |
|---|---|---|---|
| UHPHS | HHSDPA/DHSDPB(1)<br>HHSDMA/ DHSDMB(1) | Application-dependent(2)(3) | Pull-down output at reset |
| UDPHS | HHSDP/HHSDM | Application-dependent(2) | Pull-down output at reset |
| HSIC | HHSTROBE/ HHDATA | Application-dependent(2) | Pull-down output at reset |

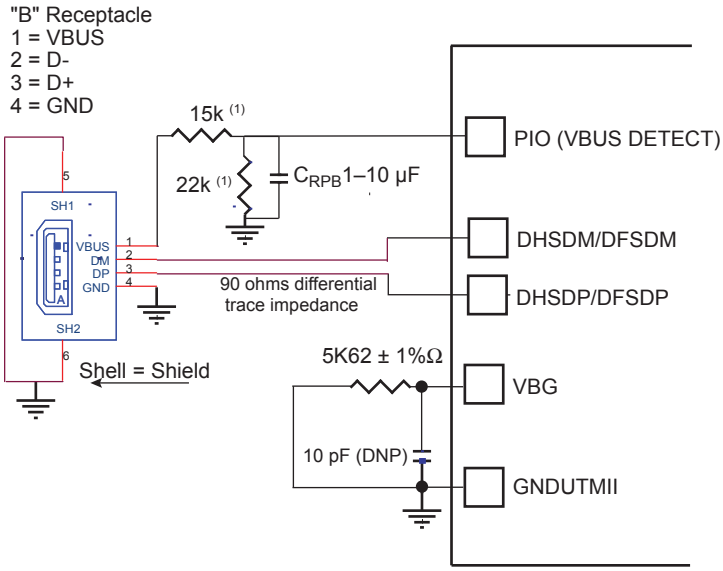
**Note:**

1. UDPHS shares Port A with UHPHS.

2. Example of a USB High Speed Host connection, refer to the section USB Host High Speed Port (UHPHS).




3. Typical USB High Speed Device connection, refer to the section USB Host High Speed Port (UHPHS).




**Note:** The values shown on the 22 kΩ and 15 kΩ resistors are only valid with 3.3V supplied PIOs.

## 68.11 Boot Program Hardware Constraints

Refer to the section Standard Boot Strategies for more details on the boot program.

## 68.12 Layout and Design Constraints

**Note:** All PIOs shared, multiplexed, connected to various components and connectors must be connected through serial resistors 22R and 39R for clock signals. The resistors must be populated as close as possible to the SAMA5D2.

### 68.12.1 General Considerations

This chapter provides routing guidelines for layout and design of a printed circuit board using high-speed interfaces, Serial, Ethernet and USB 2.0. The signal integrity rules for high-speed interfaces need to be considered. In fact, it is highly recommended that the board design be simulated to determine optimum layout for signal integrity and quality. Keep in mind that this document can only highlight the most important issues that should be considered when designing the SAMA5D2 board. The designer has to take into account the corresponding information (specification, design guidelines, etc.) contained in the documentation of all other devices that are to be implemented on board.

### 68.12.2 Considerations for High-Speed Differential Interfaces

The following recommendations for designing with high-speed differential signals are useful to maintain maximum SAMA5D2 board performance when implementing these interfaces.

- Use controlled impedance PCB traces that match the specified differential impedance.
- Keep the trace lengths of the differential signal pairs as short as possible.
- The differential signal pair traces should be trace-length matched.
- Maintain parallelism and symmetry between differential signals with the trace spacing needed to achieve the specified differential impedance.
- Maintain maximum possible separation between the differential pairs and any high-speed clocks/ periodic signals (CMOS/TTL) and any connector leaving the PCB (such as I/O connectors, control and signal headers, or power connectors).
- Route differential signals on the signal layer nearest to the ground plane using a minimum of vias and corners. This will reduce signal reflections and impedance changes. Use GND stitching vias when changing layers.
- It is best to put CMOS/TTL and differential signals on different layers which should be isolated by the power and ground planes.
- Avoid tight bends. When it becomes necessary to turn 90°, use two 45° turns or an arc instead of making a single 90° turn.
- Do not route traces under crystals, crystal oscillators, clock synthetizers, magnetic devices or ICs that use, and/or generate, clocks.
- Stubs on differential signals should be avoided due to the fact that stubs cause signal reflections and affect signal quality.
- Keep the length of high-speed clock and periodic signal traces that run parallel to high-speed signal lines at a minimum to avoid crosstalk. Based on EMI testing experience, the minimum suggested spacing to clock signals is 50 mil.
- Use a minimum of 20 mil spacing between the differential signal pairs and other signal traces for optimal signal quality. This helps to prevent crosstalk.
- Route all traces over continuous planes (VCC or GND) with no interruptions.
- Avoid crossing over anti-etch if at all possible. Crossing over anti-etch (split planes) increases inductance and radiation levels by forcing a greater loop area.

### 68.12.3 DDR Layout and Design Considerations

Refer to the document “SAMA5D2 Layout Recommendations”, document no. 44041.

### 68.12.4 e.MMC Routing

Refer to the Micron Technical Note TN-FC-35: *e.MMC PCB Design Guide*. This document is intended as guide for PCB designers using Micron e.MMC devices and discusses the primary issues affecting design and layout.

### 68.12.5 USB Trace Routing Guidelines

- Maintain parallelism between USB differential signals with the trace spacing needed to achieve 90-ohm differential impedance. Deviations will normally occur due to package breakout and routing to connector pins. Just ensure the amount and lengths of the deviations are kept to the minimum possible.
- Use an impedance calculator to determine the trace width and spacing required for the specific board stack-up being used. Example: Layer Stacking, 7.5-mil traces with 7.5-mil spacing results in approximately 90-ohm differential trace impedance.
- Minimize the length of high-speed clock and periodic signal traces that run parallel to high-speed USB signal lines, to minimize crosstalk.
- Based on EMI testing experience, the minimum suggested spacing to clock signals is 50 mils. Based on simulation data, use 20-mil minimum spacing between high-speed USB signal pairs and other signal traces for optimal signal quality. This helps to prevent crosstalk.

**Table 68-12. USB Trace Routing Guidelines**

| Parameter | Trace Routing |
|---|---|
| Signal length allowance for the SAMA5D2 | 14.0 inches<br>Valid for a damping value of the PCB trace of 0.11 dB/inch @ 0.4 GHz (common value for FR-4 based material) |
| Differential impedance | 90 ohms ±15% |
| Single-ended impedance | 45 ohms ±10% |
| Trace width (W) | 5 mils (microstrip routing) |
| Spacing between differential pairs (intra-pair) | 6 mils (microstrip routing) |
| Spacing between pairs (inter-pair) | Min. 20 mils |
| Spacing between differential pairs and high-speed periodic signals | Min. 50 mils |
| Spacing between differential pairs and low-speed non-periodic signals | Min. 20 mils |
| Length matching between differential pairs (intra-pair) | 150 mils |
| Reference plane | GND referenced preferred |
| Spacing from edge of plane | Min. 40 mils |
| Vias usage | Try to minimize the number of vias |

### 68.12.6 QSPI Pull-up Resistors

The ROM code **removes** the internal pull-up resistors when it configures PIO controller to mux the QSPI controller I/O lines. Therefore the probing step may fail if the Quad I/O mode of the memory has not been enabled yet and if this memory does not embed internal pull-up resistor on #HOLD or #RESET pin.

For this reason, it is recommended to add external pull-up resistors if needed on the four I/O data lines MOSI/IO0, MISO/IO1, #WP/IO2 and #HOLD/IO3.

Another solution is to update the Quad Enable non-volatile bit in the relevant register to reassign #WP and #HOLD/#RESET pins to functions IO2 and IO3.

### 68.12.7 Considerations for PTC Interface

Particular care must be taken during the layout of the PTC interface for the signals PTC_Xm and PTC_Yn. The PTC Debug Port (PTC_PORT_x) can be routed normally.

**X-Lines (PTC_Xm)**

- Must be routed on TOP and BOTTOM side as often as possible.
- Can be routed in inner layer when necessary.
- A maximum of 4 vias per line is allowed.
- Is possible to cross the lines if necessary.
- X-lines must be as short as possible with a maximum intrinsic capacitance of 15pF.

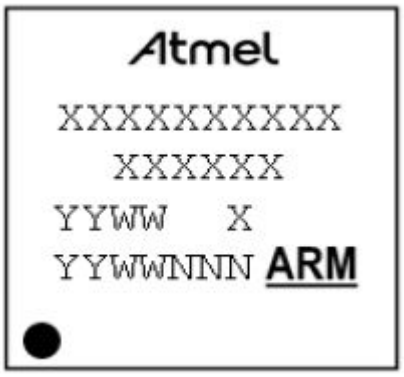
**Y-Lines (PTC_Yn)**

- Must be routed on TOP side only.
- Only one via per line is allowed.
- Never cross the Y-lines.
- Y-lines must be as short as possible with a maximum intrinsic capacitance of 15pF.

Respect an absolute clearance in PTC routing area. No ground path or plane, no power path or plane, and no signals except other PTC lines respecting the recommendations above should overlay the PTC signals in another PCB layer.

# 69. Marking

Top marking follows the scheme below:




with possible values:

| Line | Description | Values |
|---|---|---|
| 1 | Company Logo | Atmel |
| 2 | Device Name | ATSAMA5D21<br>ATSAMA5D22<br>ATSAMA5D23<br>ATSAMA5D24<br>ATSAMA5D26<br>ATSAMA5D27<br>ATSAMA5D28 |
| 3 | Device Information | CN<br>CU |
| 4 | Date Code, Design Revision | YYWW C<br>YYWW B<br>YYWW A |
| 5 | Lot Traceability, ARM logo | YYWWNNN ARM |

# 70. Ordering Information

| Ordering Code | MRL | Package | Carrier Type | Operating Temperature Range |
|---|---|---|---|---|
| ATSAMA5D21C-CU | C | TFBGA196 | Tray | -40°C to 85°C |
| ATSAMA5D21C-CUR | C | | Tape and reel | |
| ATSAMA5D22C-CN | C | | Tray | -40°C to 105°C |
| ATSAMA5D22C-CNR | C | | Tape and reel | |
| ATSAMA5D22C-CU | C | | Tray | -40°C to 85°C |
| ATSAMA5D22C-CUR | C | | Tape and reel | |
| ATSAMA5D23C-CN | C | | Tray | -40°C to 105°C |
| ATSAMA5D23C-CNR | C | | Tape and reel | |
| ATSAMA5D23C-CU | C | | Tray | -40°C to 85°C |
| ATSAMA5D23C-CUR | C | | Tape and reel | |
| ATSAMA5D24C-CU | C | TFBGA256 | Tray | -40°C to 85°C |
| ATSAMA5D24C-CUF01(1) | C | | | |
| ATSAMA5D24C-CUR | C | | Tape and reel | |
| ATSAMA5D26C-CN | C | LFBGA289 | Tray | -40°C to 105°C |
| ATSAMA5D26C-CNR | C | | Tape and reel | |
| ATSAMA5D26C-CU | C | | Tray | -40°C to 85°C |
| ATSAMA5D26C-CUR | C | | Tape and reel | |
| ATSAMA5D27C-CN | C | | Tray | -40°C to 105°C |
| ATSAMA5D27C-CNR | C | | Tape and reel | |
| ATSAMA5D27C-CU | C | | Tray | -40°C to 85°C |
| ATSAMA5D27C-CUR | C | | Tape and reel | |
| ATSAMA5D28C-CN | C | | Tray | -40°C to 105°C |
| ATSAMA5D28C-CNR | C | | Tape and reel | |
| ATSAMA5D28C-CU | C | | Tray | -40°C to 85°C |
| ATSAMA5D28C-CUR | C | | Tape and reel | |

**Note:**

1. The Boot Configuration Word reset value for ATSAMA5D24C-CUF01 is 0x00000C00, meaning that the boot on SDMMC1 and SDMMC0 are disabled. Refer to Boot Configuration Word.

# 71. Revision History

## 71.1 Revision DS60001476C

| Changes |
|---|
| **Global format change**<br>In peripheral sections:<br>- User Interface table is now called Register Summary and features new information.<br>- Details for registers have changed. The details are now Name, Offset, Reset and Property **(Note)**.<br>- Each bitfield now displays its Access and Reset value.<br>**Note**: Absolute register addresses are available in Memories. |
| Updated Block Diagram. |
| Updated Signal Description List for PTC signals. |
| Table Pin Description (all packages): added note on JTAG boundary scan. |
| **Memory Mapping** |
| In Figure 8-1, updated SYSCWP, PMC, RTC and RSTC. |
| **Debug and Test Features** |
| Updated Figure 15-2. |
| Deleted section *Chip Access Using JTAG Connection*. |
| Updated Product part number and JTAG ID Code value in Boundary JTAG ID Register. |
| Updated Product Version Number and Debug Port JTAG IDCODE in JTAG-DP Device ID Code Register. |
| **Standard Boot Strategies** |
| Updated Boot Configuration. |
| **Matrix (H64MX/H32MX)** |
| Updated table Peripheral Identifiers. |
| Master to Slave Access on H32MX: connection between ICM and H32MX removed. |
| Added note on AESB after table Master to Slave Access on H64MX. |
| Section "TrustZone Extension to AHB and APB" renamed to 18.12 TrustZone Technology and reorganized content. |
| In Principles, renamed ‘Always Non-secure’ to ‘Never Secure’ |
| **Special Function Registers (SFR)** |
| Section 19.3.2 SFR_OHCIICR: updated SUSPEND_A, SUSPEND_B and SUSPEND_C definitions |
| **Advanced Interrupt Controller (AIC)** |
| Updated Figure 21-8. |
| Updated Figure in Section 21.3 Block Diagram; updated Section 21.7.3 Interrupt Sources. |
| **Shutdown Controller (SHDWC)** |
| Added information on Write Protection. |
| **Periodic Interval Timer (PIT)** |
| Added information on Write Protection. |

| .....continued |
|---|
| **Changes** |
| **Real-time Clock (RTC)** |
| Updated content of section Waveform Generation. |
| Added information about how to clear bits/fields. |
| **System Controller Write Protection** |
| Section 27.1.2 Register Write Protection: updated list of write-protected registers. |
| **Slow Clock Controller (SCKC)** |
| Updated:<br>• Section 28.1 Description<br>• Section 28.3 Block Diagram<br>• Section 28.4 Functional Description<br>• Section 28.4.1 Switching from Embedded 64 kHz RC Oscillator to 32.768 kHz Crystal Oscillator<br>• Section 28.4.2 Switching from 32.768 kHz Crystal Oscillator to Embedded 64 kHz RC Oscillator<br>• Section 28.5.1 SCKC_CR |
| **Power Management Controller (PMC)** |
| Clock Assignments: removed LCDC row. |
| Removed content related to the CKGR_UCKR.BIASEN bit in Section 32.19 Programming Sequence and Section 32.22.7 CKGR_UCKR |
| Updated Section 32.9 Peripheral and Generic Clock Controller. |
| **Parallel Input/Output Controller (PIO)** |
| Updated Figure 33-2. |
| **External Memories** |
| Table 34-2: Added note on connecting DDR_DQSN. |
| **AHB Multiport DDR-SDRAM Controller (MPDDRC)** |
| Embedded Characteristics: added “Write Leveling not supported” and added “Low-cost LPDDR1with 2 internal banks” and “Arbitration policies” characteristics |
| Updated Block Diagram. |
| DDR2-SDRAM Initialization: added Step 2. |
| Low-power DDR2-SDRAM Initialization: added Step 2. |
| DDR3-SDRAM/DDR3L-SDRAM Initialization: added Step 2. |
| Restored title of Refresh (Autorefresh Command). |
| Scrambling/Unscrambling Function: MPDDRC_KEY1/2 replaced with KEY1/2. |
| Added Monitor. |
| Updated Sequential Mapping DDR-SDRAM Configuration Mapping: 8K Rows, 512/1024 Columns, 8 banks and Interleaved Mapping DDR-SDRAM Configuration Mapping: 8K Rows, 512/1024 Columns, 8 banks. |
| MPDDRC Mode Register: updated DAI bit description. |
| MPDDRC Configuration Register: updated UNAL bit description. |
| MPDDRC Memory Device Register: updated WL, RL3, MANU_ID, REV_ID, TYPE, DENSITY and IO_WIDTH bit descriptions. |
| MPDDRC Low-power DDR2 Low-power DDR3 and DDR3 Timing Calibration Register: updated RZQI bit description. |
| MPDDRC I/O Calibration Register: updated CALCODEP and CALCODEN field descriptions. |
| MPDDRC Read Data Path Register: updated SHIFT_SAMPLING field description. |

| .....continued |
|---|
| **Changes** |
| MPDDRC Monitor Configuration Register: updated REFR_CALIB bit description. |
| **DMA Controller (XDMAC)** |
| Section 37.5.1 Basic Definitions: added "Stride" definition; added 37.5.2 Data Striding Diagram. |
| Updated Section 37.8 XDMAC Software Requirements;Section 37.9.28 XDMAC_CC: updated PROT bit description. |
| **Ethernet MAC (GMAC)** |
| MAC Transmit Block: removed reference to non-existing bit HDFC. |
| GMAC Network Status Register: updated reset value for GMAC_NSR. |
| **USB High Speed Device Port (UDPHS)** |
| Updated Endpoint Configuration and DPRAM Management. |
| Removed figure Example of DPRAM Allocation and Reorganization. |
| **USB Host High Speed Port (UHPHS)** |
| Updated Section 41.3 Block Diagram, Section 41.5.1 I/O Lines, Section 41.6.2 EHCI, and Section 41.6.3 OHCI. |
| Power Management: removed reference to bit BIASEN (UTMI BIAS Enable). |
| Updated Figure 41-2. |
| Added Section 41.6.1 UTMI Transceivers Sharing. |
| Section 41.7.4 UHPHS_USBCMD: USBCMD reset value set to 0x00080B00 (instead of "0x00080000 or 0x00080B00"). |
| Section 41.7.11 UHPHS_PORTSCx: UHPHS_PORTSCx reset value set to 0x00002000 (instead of "0x00002000 or 0x00003000"). |
| Deleted registers UHPHS_INSNREG00 to UHPHS_INSNREG05. |
| **Two-wire Interface (TWIHS)** |
| Table 45-1: START byte support now available. Updated Note 2. |
| Replaced sections "Master Mode FIFOs" and "Slave Mode FIFOs" by a single Section “FIFOs”. |
| Section FIFOs, sub-section Overview, the description for FIFO disable/enable has been changed. |
| **Flexible Serial Communication Controller (FLEXCOM)** |
| Section 46.10.7 FLEX_US_MR (SPI_MODE): restored CHMODE field and updated CHMODE description. |
| Section 46.7.3.1 Transmitter Operations: updated Figure Transmitter Status |
| Section 46.10.55 FLEX_SPI_FMR: updated RXRDYM field description |
| Updated 46.7.4.1 ISO7816 Mode Overview |
| Section 46.7.6 RS485 Mode: updated Figure Example of RTS Drive with Timeguard |
| TWI Clock Waveform Generator Register: deleted CKSRC description. |
| **Universal Asynchronous Receiver Transmitter (UART)** |
| Section47.3 Block Diagram: RXD/TXD pins were renamed to URXD/UTXD. |
| Section47.5.7 Register Write Protection: added UART Receiver Time-out Register to the list. |
| **Quad Serial Peripheral Interface (QSPI)** |
| QSPI Bus Clock Modes: updated column “Capture QSCK Edge”. |
| **Secure Digital MultiMedia Card Controller (SDMMC)** |
| SDMMC Preset Value Register: deleted Note from SDMMC_PVRx register description. |

| .....continued |
|---|
| **Changes** |
| SDMMC Maximum Current Capabilities Register: corrected access to Read/Write. Added note on CAPWREN below register table. |
| SDMMC Capabilities Control Register: updated CAPWREN bit description. |
| **Image Sensor Controller (ISC)** |
| Section51.6.1 ISC_CTRLEN: added bit FUPPRO at index 9 and bit description. |
| Section51.6.30 ISC_GAM_CTRL: added bit BIPART at index 4 and bit description. |
| **Controller Area Network (MCAN)** |
| Description, Section 52.2 Embedded Characteristics: added information on ISO non-compliance |
| **Pulse Width Modulation Controller (PWM)** |
| PWM Clock Generator: updated figure. |
| **Secure Fuse Controller (SFC)** |
| Number of maskable Data Registers modified to 8. Number of Data Registers modified to 17. |
| Modified ACE bit name in Section 56.5.3 SFC_IER, Section 56.5.4 SFC_IDR, Section 56.5.5 SFC_IMR and Section 56.5.6 SFC_SR. |
| **Integrity Check Monitor (ICM)** |
| Updated Section 57.5.4.2 Processing Period. |
| **Advanced Encryption Standard (AES)** |
| Section 59.1 Description: added XTS to supported modes. |
| Section 59.4.2 Operating Modes: added bullet for XTS mode; added bullet for CBC-MAC. |
| Last Output Data Mode: Added (CBC-MAC) to section title. Added details on CMAC algorithm. |
| Section59.4.7.1 Manual Mode: table "Authorized Input Data Registers": updated for XTS algorithm. |
| Section 59.4.7.3 DMA Mode: table "DMA Data Transfer Type for the Different Operating Modes": updated for XTS algorithm. |
| IPSEC Padding: added detail on padding length. |
| SSL Padding: added detail on incremental integer values. |
| Updated DMA Mode. |
| Updated If AES_MR.LOD = 1. |
| **Triple Data Encryption Standard (TDES)** |
| Last Output Data Mode (CBC-MAC): added “CBC-MAC” in section title and added second paragraph. |
| Updated Section 61.4.2.3 DMA Mode and Section 61.4.3.2.2 TDES_MR.LOD = 1. |
| **Analog-to-Digital Controller (ADC)** |
| ADC Channel Status Register: updated CHx bit description |
| Updated USCHx field description in ADC Channel Sequence 1 Register and ADC Channel Sequence 2 Register |
| **Electrical Characteristics** |
| In DC Characteristics updated:<br>- Vhyst parameter<br>- VDDIODDR<br>- conditions for VOL and VOH |
| Updated Section 66.5.2 Backup Mode with DDR in Self-Refresh. |
| Master Clock Waveform Parameters: changed VDDIODDR range. |

| .....continued |
|---|
| **Changes** |
| Corrected references to CKGR_MOR bits and added VIL_XIN and VIH_XIN parameters in XIN Clock Electrical Characteristics. |
| Updated 12 MHz RC Oscillator Characteristics. |
| Updated PLLA Characteristics. |
| Added ADC Bias Current. |
| Added Pen Detect Characteristics. |
| Updated Analog Comparator Characteristics. |
| Two Wire Serial Bus Requirements: corrected definition of TWCK high and low periods. |
| DDR2-SDRAM System Clock Waveform Parameters: VDDIODDR range changed. |
| LPDDR1-SDRAM System Clock Waveform Parameters: VDDIODDR range changed. |
| Section 66.21 SSC Timings: added SSC14 and SSC15 timings |
| **Schematic Checklist** |
| ICE and JTAG: added note on JTAG boundary scan. |
| **Errata** |
| Removed from data sheet to create separate document, *SAMA5D2 Family Silicon Errata and Data Sheet Clarification*, document number DS80000803. |

## 71.2 Revision DS60001476B

| Changes |
|---|
| **Section 6. “Package and Pinout”** |
| Table 6-2 Pin Description: added notes (2) and (3). |
| Table 6-3 Pin Description (SAMA5D23 pins different from those in Table 6-2 “Pin Description”): added note (2). |
| Table 6-4 Pin Description (SAMA5D28B/C pins different from those in Table 6-2 “Pin Description”): added note (2). |
| **Section 7. “Power Considerations”** |
| Updated Figure 7-1 “Recommended Powerup Sequence” and Table 7-2 “Powerup Timing Specification”. |
| **Section 16. “Standard Boot Strategies”** |
| Updated Section 16.4.7.3 “SDCard / e.MMC Boot”. |
| Reworked Section 16.4.7.6 “QSPI NOR Flash Boot for MRL C”. |
| **Section 29. “Peripheral Touch Controller (PTC)”** |
| - Replaced "Peripheral Touch Controller" with "Peripheral Touch Controller Subsystem". |
| - Changed "PTC_IRQ_EVT" to "PTC_IRQ". |
| - Replaced "PIO" and "IO" with "GPIO". |
| Updated: |
| - Figure 29-1. "PTC Block Diagram" |
| - Section 29.2 “Embedded Characteristics” |
| - Section 29.5.2 “I/O Lines” |
| - Section 29.5.3 “Interrupt Sources” |

| .....continued |
|---|
| **Changes** |
| - Section 29.6.2.3 “Firmware in SRAM Code Area” |
| Section 29.5.1 “Power Management”: updated description of “SLCK”. |
| Section 29.6.3.1 “PTC Digital Controller Operations”: modified Sensing mode description. |
| Renamed and reworked Section 29.7.1 “PTC Command Register” and Section 29.7.2 “PTC Interrupt Status Register”. |
| Section 29.7.3 “PTC Enable Register”: removed field CLR_IRQEN, and replaced field SET_IRQEN with bits IER0, IER1, IER2, IER3. |
| Renamed the following registers: |
| - “Mode 1: Write Access” to “PTC Command Register” |
| - “Host Flags” to “PTC Interrupt Status Register” |
| - “Host Flags Control” to “PTC Enable Register” |
| Section 39. “LCD Controller (LCDC)” |
| Corrected Figure 39-1. Block Diagram (added “PP Layer” block). |
| **Section 66. “Electrical Characteristics”** |
| Added Section 66.10 ”PTC Characteristics”. |
| Section 66.5.3.1 “ULP0 Mode”: updated steps (5) and (6). |
| Table 66-9 “Typical Peripheral Power Consumption by Peripheral in Active Mode”: corrected UHPHS consumption value from "12 *MCK + 4900*DR" to "12 *MCK + 490*DR" and UDPHS consumption value from “10 *MCK + 2060*DR” to “10 *MCK + 206*DR” |
| Table 66-13 “VDDCORE Power Consumption in Ultra Low-power Mode: AMP2”: removed ULP0 512 Hz row. |
| **Section 68. “Schematic Checklist”** |
| Removed Section 68.3 “Shutdown Considerations” and Section 68.4 “Wakeup Considerations” (redundant with content in Section 7. “Power Considerations”). |
| Table 68-1 “Power Supply Connections”: updated VDDCORE recommended pin connection from “1.08V to 1.32V” to “1.1V to 1.32V”. |
| Table 68-2 “Clock, Oscillator and PLL Connections”: updated main oscillator recommended pin connection from "Crystals between 8 and 16 MHz" to "Crystals between 8 and 24 MHz". |
| Table 68-6 “Reset and Test Connections”: updated NRST recommended pin connection. |
| **Section 69. “Marking”** |
| Updated marking information. |
| **Section 71. “Errata”** |
| Added the following issues in Section 71.1 “Errata - SAMA5D2 MRL C Parts”: |
| Section 71.1.9 “Master CAN-FD Controller (MCAN)” to Section 71.1.18 “MCAN High Priority Message (HPM)” and Section 71.1.19 “ROM Code: Using JTAG IOSET 4”. |
| Added the following issues in Section 71.2 “Errata - SAMA5D2 MRL B Parts”: |
| Section 71.2.10 “Master CAN-FD Controller (MCAN)” to Section 71.2.19 “MCAN High Priority Message (HPM)” and Section 71.2.20 “ROM Code: Using JTAG IOSET 4”. |
| Added the following issues in Section 71.3 “Errata - SAMA5D2 MRL A Parts”: |
| Section 71.3.26 “Master CAN-FD Controller (MCAN)” to Section 71.3.35 “MCAN High Priority Message (HPM)” and Section 71.3.36 “ROM Code: Using JTAG IOSET 4”. |

## 71.3 Revision DS60001476A

| Changes |
|---|
| General |
| - Template update: Moved from Atmel to Microchip template. |
| - The datasheet is assigned a new document number (DS60001476) and revision letter is reset to A. |
| --- Document number DS60001476 revision A corresponds to what would have been 11267 revision F. |
| - ISBN number assigned. |
| **“Features”**: added PTC. |
| **Table 2. “Configuration Summary”**: added PTC. Corrected number of Timers. |
| **Section 3. “Block Diagram”** |
| Figure 3-1 “SAMA5D2 Series Block Diagram”: corrected SDMMC signals. Updated peripheral bridge naming. Added PTC. Removed signal names. |
| **Section 4. “Signal Description”** |
| Table 4-1 “Signal Description List”: renamed SDMMCx_VDDSEL to SDMMCx_1V8SEL. Added PTC pins on PD0 to PD18. |
| Added **Section 5. “Safety and Security Features”**. |
| **Section 6. “Package and Pinout”** |
| Added Note on IO sets. |
| Table 6-2 “Pin Description”: for P15/R14 renamed SDMMC0_VDDSEL to SDMMC0_1V8SEL. |
| **Section 7. “Power Considerations”** |
| Table 7-1 “SAMA5D2 Power Supplies”: in VDDBU row, corrected RC Oscillator frequency to 64 kHz. |
| **Section 8. “Memories”** |
| Figure 8-1 “Memory Mapping”: renamed MATRIXx blocks. Added PTC. Renamed TC blocks. Added SYSCWP block. |
| **Section 11. “Peripherals”** |
| Table 11-1 “Peripheral Identifiers”: corrected reference to SDMMC. Assigned ID 58 to Peripheral Touch Controller (PTC). |
| Section 11.4 “Peripheral Clock Types”: removed clock type HCLOCK and PCLOCK from table. |
| **Section 12. “Chip Identifier (CHIPID)”** |
| Table 12-1 “SAMA5D2 Chip ID Registers”: added chip ids for MRL C revision. |
| **Section 13. “ARM Cortex-A5”** |
| Section 13.4.7.3 “Debug”: updated Note. |
| **Section 16. “Standard Boot Strategies”** |
| Replaced all occurrences of “Spansion” by “Cypress”. |
| Updated Section 16.3 “Chip Setup”. |
| Updated Section 16.4.1 “Boot Configuration Word”. |
| Section 16.4.2 “Boot Sequence Controller Configuration Register”: updated BUREG_VALID bit description. |
| Added Note to descriptions of SDMMC_0 and SDMMC_1 in Section 16.4.4 “Boot Configuration Word”. |
| Updated Section 16.4.5 “NVM Boot Sequence” and figures. |
| Renamed Section 16.4.7.5 “QSPI Flash Boot for MRL A and MRL B”. |

| .....continued |
|---|
| **Changes** |
| Added Section 16.4.7.6 “QSPI Flash Boot for MRL C”. |
| Updated Section 16.4.8 “Hardware and Software Constraints”: |
| -- added Table 16-5 “Clock Frequencies during External Memory Boot Sequence”. |
| -- Table 16-6 “PIO Driven during Boot Program Execution”: renamed SDMMC_VDDSEL to SDMMC0_1V8SEL. Added column “Drive Strength (MRL C only)”. |
| Section 16.5 “SAM-BA Monitor”: deleted sentence on Main Clock; updated 3rd paragraph. |
| Updated Section 16.6.1 “Fuse Bit Mapping”. |
| **Section 18. “Matrix (H64MX/H32MX)”** |
| Table 18-5 “List of H32MX Slaves”: added Peripheral Touch Controller (PTC) at Slave 6. |
| Table 18-6 “Master to Slave Access on H32MX”: added Peripheral Touch Controller (PTC) at Slave 6. |
| Table 18-9 “Peripheral Identifiers”: added PTC at ID 58. |
| **Section 19. “Special Function Registers (SFR)”** |
| Section 19.3.5 “UTMI Clock Trimming Register”: VBG now 2 bits wide at index [17:16] (was [19:16]). |
| **Section 20. “Special Function Registers Backup (SFRBU)”** |
| Section 20.3.3 “SFRBU DDR BU Mode Control Register” changed occurrences of VCCCORE to VDDCORE. |
| **Section 26. “Real-time Clock (RTC)”** |
| Table 26-1 “Register Mapping”: updated offsets as of 0xCC. Deleted RTC_WPMR at offset 0xE4. |
| Deleted “Section 25.6.23 RTC Write Protection Mode Register”. |
| Added **Section 27. “System Controller Write Protection (SYSCWP)”.** |
| Added **Section 29. “Peripheral Touch Controller (PTC)”.** |
| **Section 33. “Power Management Controller (PMC)”** |
| Reorganized order of sub-sections within the chapter. |
| Updated Figure 33-2 “H32MX 32-bit Matrix Clock Configuration”. |
| Figure 33-1 “General Clock Block Diagram”: updated PMC_PCR block. |
| Added Section 33.8 “Core and Bus Independent Clocks for Peripherals”. |
| Added Section 33.9 “Peripheral and Generic Clock Controller”. |
| Deleted section “Peripheral Clock Controller”. |
| Deleted section “Generic Clock Controller”. |
| Figure 33-10 “Clock Failure Detection (Example)”: corrected CDFEV to CFDEV and CDFS to CFDS. |
| Section 33.19 “Programming Sequence”: deleted paragraph on DIVA from Step 6. |
| Section 33.22.10 “PMC Clock Generator PLLA Register”: changed DIVA description for value ‘0’. |
| **Section 35. “External Memories”** |
| Updated Figure 35-2 “MPDDRC Block Diagram”. |
| Table 35-1 “DDR/LPDDR I/O Lines Description”: DDR_DQS[3:0] and DDR_DQSN[3:0] now type I/O. |
| Aligned signal names in schematics of Section 35.1 “Multiport DDR-SDRAM Controller (MPDDRC)” with signal names in Table 35-2 “I/O Lines Usage vs Operating Modes”. |
| **Section 36. “Multiport DDR-SDRAM Controller (MPDDRC)”** |

| .....continued |
|---|
| **Changes** |
| Section 36.1 “Description”: corrected supported CAS latency. |
| Section 36.4.1 “Low-power DDR1-SDRAM Initialization”: updated Step 12. |
| Section 36.4.2 “DDR2-SDRAM Initialization”: updated Step 22. |
| Section 36.4.3 “Low-power DDR2-SDRAM Initialization”: updated Step 22. |
| Section 36.4.4 “DDR3-SDRAM/DDR3L-SDRAM Initialization”: updated Step 13. |
| Section 36.4.5 “Low-power DDR3-SDRAM Initialization”: updated Step 22. |
| Table 36-1 “CAS Write Latency”: added row for Low-power DDR3-SDRAM. Corrected typo in note. |
| Table 36-2 “CAS Read Latency”: added row for Low-power DDR3-SDRAM. Corrected typo in note. |
| Section 36.7.2 “MPDDRC Refresh Timer Register”: updated the method to compute COUNT. |
| Section 36.7.3 “MPDDRC Configuration Register”: updated NC field description table. |
| Section 36.7.6 “MPDDRC Timing Parameter 2 Register”: in TRPA description, added “In the case of LPDDR2-SDRAM, this field is equivalent to tRPAB.” |
| Section 36.7.10 “MPDDRC Low-power DDR2 Low-power DDR3 and DDR3 Calibration and MR4 Register”: in COUNT_CAL field description, added ‘One ZQCS command can effectively correct at least 1.5% of output impedance errors within Tzqcs.’ and ‘TSens and VSens are given by the manufacturer (Output Driver Sensitivity definition). Tdriftrate and Vdriftrate are defined by the end user.’ |
| **Section 38. “DMA Controller (XDMAC)”** |
| Added information regarding XDMAC_CC.INITD in Section 38.8 “XDMAC Software Requirements” and Section 38.9.28 “XDMAC Channel x [x = 0..15] Configuration Register”. |
| Section 38.9.3 “XDMAC Global Weighted Arbiter Configuration Register”: replaced “XDMAC scheduler” with “DMAC scheduler” throughout. |
| **Section 39. “LCD Controller (LCDC)”** |
| Standardized signal names from ‘LCD_XXX’ to ‘LCDXXX’ (‘underscore’ character removed) |
| Section 39.2 “Embedded Characteristics”: removed “(at synthesis time)” from characteristic “Asynchronous Output Mode Supported”. |
| **Section 40. “Ethernet MAC (GMAC)”** |
| Section 40.2 “Embedded Characteristics”: deleted queue sizes (now found in Table 40-6 “Queue Size”). |
| Table 40.6.3.9 “Priority Queueing in the DMA”: added Table 40-6 “Queue Size” and updated queue sizes. |
| Section 40.6.15 “Timestamp Unit”: changed pin reference from “TIOB11/PD22” to “TIOA11/PD21”. |
| Added Section 40.6.18 “Energy-efficient Ethernet Support” |
| Updated Section 40.6.19 “802.1Qav Support - Credit-based Shaping”: added definitions of portTransmitRate and IdleSlope; updated content on queue priority management. |
| Added Section 40.6.20 “LPI Operation in the GMAC”. |
| Table 40-18 “Register Mapping”: added registers at offsets 0x270 to 0x27C. |
| Section 40.8.1 “GMAC Network Control Register”: added bit 19: TXLPIEN: Enable LPI Transmission (was ‘reserved’). Added bit description. Changed description of SRTSM bit. |
| Section 40.8.3 “GMAC Network Status Register”: added bit 7: RXLPIS: LPI Indication (was ‘reserved’). Added bit description. |
| Added bit 27: RXLPISBC: Receive LPI indication Status Bit Change and bit description and added bit 29: TSUTIMCOMP: TSU timer comparison interrupt and bit description in: |
| - Section 40.8.10 “GMAC Interrupt Status Register” |
| - Section 40.8.11 “GMAC Interrupt Enable Register” |
| - Section 40.8.12 “GMAC Interrupt Disable Register” |

| .....continued |
|---|
| **Changes** |
| - Section 40.8.13 “GMAC Interrupt Mask Register” |
| Section 40.8.13 “GMAC Interrupt Mask Register”: added bit 26, SRI, and bit 28, WOL, and bit descriptions. |
| Added following sections: |
| - Section 40.8.106 “GMAC Received LPI Transitions” |
| - Section 40.8.107 “GMAC Received LPI Time” |
| - Section 40.8.108 “GMAC Transmit LPI Transitions” |
| - Section 40.8.109 “GMAC Transmit LPI Time” |
| Section 40.8.115 “GMAC Credit-Based Shaping IdleSlope Register for Queue A” and Section 40.8.116 “GMAC Credit-Based Shaping IdleSlope Register for Queue B”: updated example for calculation of IdleSlope. |
| **Section 41. “USB High Speed Device Port (UDPHS)”** |
| Table 41-6 “Register Mapping”: offsets 0xD0 to 0xDC now ‘reserved’. |
| Deleted internal registers: |
| - UDPHS Test SOF Counter Register |
| - UDPHS Test A Counter Register |
| - UDPHS Test B Counter Register |
| - UDPHS Test Mode Register |
| **Section 43. “Audio Class D Amplifier (CLASSD)”** |
| Section 43.6.6 “Application Schematics For Use Case Examples”: for Use Case 1, added information on external MOSFET selection. |
| Section 44. “Inter-IC Sound Controller (I2SC)” |
| In text, tables and figures, pin names changed to: |
| - I2SC_MCK |
| - I2SC_CK |
| - I2SC_WS |
| - I2SC_DI |
| - I2SC_DO |
| Updated Figure 44-1 “I2SC Block Diagram”. |
| Section 44.6.1 “Initialization”: added detail on configuring SFR_I2SCLKSEL. |
| Section 44.6.5 “Serial Clock and Word Select Generation”: updated paragraph on I2SC input clock selection in Master mode. |
| Updated Figure 44-3 “I2SC Clock Generation”. |
| Updated figures in Section 44.7 “I2SC Application Examples”. |
| Section 44.8.2 “I2SC Mode Register”: updated MODE bit description for value ‘1’. Updated IMCKDIV and IMCKMODE field descriptions. |
| **Section 46. “Two-wire Interface (TWIHS)”** |
| Added Section 46.7.2 “TWIHS Control Register (FIFO_ENABLED)”. |
| Added Section 46.7.8 “TWIHS Status Register (FIFO_ENABLED)”. |
| **Section 47. “Flexible Serial Communication Controller (FLEXCOM)”** |
| Corrected Figure 47-27 “RTS Line Software Control when FLEX_US_MR.USART_MODE = 2”. |
| Reworked Section 47.7.11 “USART FIFOs”. |

| .....continued |
|---|
| **Changes** |
| Updated Section 47.8.3.5 “Peripheral Selection”. |
| Reworked Section 47.8.7 “SPI FIFOs”. |
| Section 47.9.3.9 “SMBus Mode”: corrected typo in SMBEN bit name (was ‘SMEN’). |
| Created Section 47.9.6 “TWI FIFOs” by merging Section 10.3.15 “TWI Master Mode FIFOs” and Section 10.5.9 “TWI Slave Mode FIFOs” |
| Section 47.10.43 “SPI Control Register”: updated TXFCLR and RXFCLR bit descriptions. |
| Section 47.10.50 “SPI Status Register”: updated RDRF and TDRE bit descriptions. |
| Section 47.10.55 “SPI FIFO Mode Register”: updated TXRDYM description for value ‘1’. Deleted row for value ‘2’. Updated RXRDYM description for value ‘1’ and for value ‘2’. |
| Added Section 47.10.61 “TWI Control Register (FIFO_ENABLED)”. |
| Added Section 47.10.67 “TWI Status Register (FIFO ENABLED)”. |
| **Section 49. “Serial Peripheral Interface (SPI)”** |
| Section 49.7.3.5 “Peripheral Selection”: modified sub-section “Variable Peripheral Select Mode”. |
| Section 49.7.5 “SPI Comparison Function on Received Character”: replace ‘When the CMPMODE bit is cleared in SPI_CMPR’ with ‘When SPI_MR.CMPMODE is cleared’ |
| In Section 49.7.7 “FIFOs” updated: |
| - Section 49.7.7.1 “Overview” |
| - Section 49.7.7.2 “Sending Data with FIFO Enabled” |
| - Section 49.7.7.3 “Receiving Data with FIFO Enabled” |
| - Section 49.7.7.5 “TXEMPTY, TDRE and RDRF Behavior” |
| - Section 49.7.7.6 “Single Data Mode” |
| - Section 49.7.7.7 “Multiple Data Mode” |
| - “TDRE and RDRF Configuration” |
| Section 49.8.1 “SPI Control Register”: updated TXFCLR and RXFCLR bit descriptions. |
| Section 49.8.8 “SPI Status Register”: updated RDRF and TDRE bit descriptions. |
| Section 49.8.13 “SPI FIFO Mode Register”: updated TXRDYM description for value ‘1’. Deleted row for value ‘2’. Updated RXRDYM description for value ‘1’ and for value ‘2’. |
| Updated “DMAC” in Section 49.7.7.7 “Multiple Data Mode”. |
| **Section 50. “Quad Serial Peripheral Interface (QSPI)”** |
| Section 50.1 “Description”: added Note on device support. |
| Removed references to Double Data Rate (DDR) in Section 50.2 “Embedded Characteristics” and Section 50.6.5.2 “Instruction Frame Transmission”. |
| Section 50.6.5 “QSPI Serial Memory Mode”: updated text on data transfer constraint. |
| Updated Figure 50-8 “Instruction Frame”. |
| Figure 50-9 “Instruction Transmission Flow Diagram”: |
| - Corrected typos: |
| --- “Wait for flag QSPI_SR.INSTRE ... “ (was “QSPI_CR“) |
| --- “Wait for flag QSPI_SR.CSR ... “ (was “QSPI_CR“) |
| - Added new instruction: “Read QSPI_SR (dummy read) to clear QSPI_SR.INSTRE and QSPI_SR.CSR“ |

| .....continued |
|---|
| **Changes** |
| Updated Figure 50-10 “Continuous Read Mode”, Figure 50-16 “Instruction Transmission Waveform 6”, Figure 50-17 “Instruction Transmission Waveform 7” and Figure 50-18 “Instruction Transmission Waveform 8”. |
| **Section 51. “Secure Digital MultiMedia Card Controller (SDMMC)”** |
| Section 51.2 “Embedded Characteristics”: updated bullets on support for MMC at default speed and at high speed. |
| Section 51.3 “Embedded Features for SDMMC0 and SDMMC1”: updated information on SDMMC1. |
| Section 51.10.1.3 “Boot Procedure, ADMA Mode”: removed note after step j. |
| Section 51.13.2 “SDMMC Block Size Register” updated BLKSIZE field description. |
| Section 51.13.9 “SDMMC Present State Register”: updated BUFWREN field description. |
| Section 51.13.16 “SDMMC Clock Control Register”: updated SDCLKFSEL field description. |
| Section 51.13.17 “SDMMC Timeout Control Register”: updated equation in DTCVAL field description. |
| Section 51.13.18 “SDMMC Software Reset Register”: in SWRSTALL field description, updated list of registers cleared to 0 . |
| Section 51.13.21 “SDMMC Error Interrupt Status Register (SD_SDIO)”: in CURLIM bit description, corrected reference to SDMMC_PCR (was SDMMC_PSR). |
| Section 51.13.22 “SDMMC Error Interrupt Status Register (e.MMC)”: updated ACMD bit description. |
| Section 51.13.31 “SDMMC Auto CMD Error Status Register”: changed register access to Read-only. |
| Section 51.13.32 “SDMMC Host Control 2 Register (SD_SDIO)”: corrected typo in bit 7 name (is SCLKSEL; was SLCKSEL). Updated VS18EN bit description. |
| Section 51.13.33 “SDMMC Host Control 2 Register (e.MMC)”: corrected typo in bit 7 name (is SCLKSEL; was SLCKSEL). |
| Section 51.13.41 “SDMMC Preset Value Register”: in Table 51-8 “Preset Value Register Select Condition”: corrected HSEN value in row for High Speed. |
| Section 51.13.45 “SDMMC e.MMC Control 1 Register”: in DDR bit description Note, replaced ‘HSEN’ with ‘DDR’. |
| Section 51.13.51 “SDMMC Retuning Counter Value Register”: updated description for TCVAL. |
| **Section 53. “Controller Area Network (MCAN)”** |
| Section 53.1 “Description”: updated information on compliance. |
| Section 53.4.4 “Address Configuration”: added cross-reference for clarity. |
| Section 53.5.1.4 “Transmitter Delay Compensation”: changed NTSEG1 to TSEG1. |
| **Section 54. “Timer Counter (TC)”** |
| Table 54-1 “Timer Counter Clock Assignment”: added Note below table. |
| Section 54.6.16.2 “Input Preprocessing”: removed unit following equation in 3rd paragraph. Added limitation on maximum pulse duration. |
| Section 54.6.16.4 “Position and Rotation Measurement”: in 3rd paragraph, added “The process must be started by configuring TC_CCR.CLKEN and TC_CCR.SWTRG.” |
| Section 54.6.16.6 “Detecting a Missing Index Pulse”: corrected value of TC_RC0.RC in example in 2nd paragraph. |
| Added Section 54.6.16.7 “Detecting Contamination/Dust at Rotary Encoder Low Speed”. |
| Section 54.7.16 “TC Block Mode Register”: updated MAXFILT field description. |
| In Section 54.7.17 “TC QDEC Interrupt Enable Register”, Section 54.7.18 “TC QDEC Interrupt Disable Register” and Section 54.7.19 “TC QDEC Interrupt Mask Register”: at index 3, added bit MPE and bit description. |
| Corrected occurrences of ‘MAXMP’ to ‘MAXCMP’ in Section 54.7.19 “TC QDEC Interrupt Mask Register” and in Section 54.7.20 “TC QDEC Interrupt Status Register” |
| **Section 56. “Pulse Width Modulation Controller (PWM)”** |

| .....continued |
|---|
| **Changes** |
| Section 56.6.2.2 “Comparator”: corrected ‘CRPD’ to ‘CPRD’ in formulae. |
| Table 56-8 “Register Mapping”: modfied offsets for “PWM External Trigger Register 1”, “PWM Leading-Edge Blanking Register 1”, “PWM External Trigger Register 2” and “PWM Leading-Edge Blanking Register 2”. |
| Section 56.7.43 “PWM Channel Period Register”: corrected ‘CRPD’ to ‘CPRD’ in CPRD description. |
| Section 56.7.44 “PWM Channel Period Update Register”: corrected ‘CRPDUPD’ to ‘CPRDUPD’ in CPRDUDP description. |
| **Section 57. “Secure Fuse Controller (SFC)”** |
| Removed all occurrences of ‘Atmel reserved area’ ( now just ‘reserved area’). |
| Modified bit names for APLE and ACE in: |
| - Section 57.5.3 “SFC Interrupt Enable Register” |
| - Section 57.5.4 “SFC Interrupt Disable Register” |
| - Section 57.5.5 “SFC Interrupt Mask Register” |
| - Section 57.5.6 “SFC Status Register” |
| **Section 58. “Integrity Check Monitor (ICM)”** |
| Updated Section 58.5.5 “ICM Automatic Monitoring Mode”. |
| Section 58.6.1 “ICM Configuration Register”: updated ASCD description. |
| **Section 59. “Advanced Encryption Standard Bridge (AESB)”** |
| Section 59.1 “Embedded Characteristics”: replaced "12 clock cycles encryption/decryption processing time with a 128-bit cryptographic key" with "10 clock cycles encryption/decryptioninherent processing time". |
| In Section 59.3.6 “Automatic Bridge Mode”, updated Section 59.3.6.1 “Description”. |
| **Section 60. “Advanced Encryption Standard (AES)”** |
| Replaced references to “keys” and to “AES_KEYWRx registers” with “AES Key Registers”. |
| Section 60.1 “Description”: corrected index of AES_KEYWR0 registers from 3 to 7. |
| Section 60.2 “Embedded Characteristics” replaced “12/14/16 Clock Cycles Encryption/Decryption Processing Time” with “10/12/14 Clock Cycles Encryption/Decryption Inherent Processing Time” |
| **Section 61. “Secure Hash Algorithm (SHA)”** |
| Corrected typos in some figures. |
| Section 61.4.9.1 “Manual Mode”: updated Step 2. |
| Section 61.5.2 “SHA Mode Register”: updated SMOD bit description. |
| **Section 62. “Triple Data Encryption Standard (TDES)”** |
| Replaced references to “TDES_KEYxWRx registers” with “Key Registers”. |
| **Section 63. “True Random Number Generator (TRNG)”** |
| Section 63.6.1 “TRNG Control Register”: changed field name to WAKEY (was KEY). |
| Added **Section 64. “Security Module (SECUMOD)”.** |
| **Section 65. “Analog-to-Digital Converter (ADC)”** |
| Section 65.5.3 “I/O Lines”: “ADC_ADTRG” corrected to “ADTRG”. |
| Section 65.6.14 “Automatic Error Correction”: added information about ADCMODE field. |
| Table 65-7 “ADC Running Modes”: updated Offset Error row. |

| .....continued |
|---|
| **Changes** |
| Section 65.6.18 “Fault Event”: reworked and renamed section (was previously “Fault Output”). |
| Section 65.7.2 “ADC Mode Register”: at index 30, added bit MAXSPEED and bit description |
| Section 65.7.12 “ADC Interrupt Status Register”: updated LCCHG bit description. |
| Section 65.7.20 “ADC Analog Control Register”: added Note (1). |
| **Section 66. “Electrical Characteristics”** |
| Table 66-3 “DC Characteristics”: updated min and max valued for low-level and high-level input currents (all pads). Changed ISI_MCK to ISC_MCK. |
| Updated Table 66-8 “Typical Peripheral Power Consumption by Peripheral in Active Mode” with new column “Clock”. |
| Table 66-10 “Power Consumption in Active Mode: AMP2”: Removed columns DMIPS and CoreMark. |
| In Section 66.14 “FLEXCOM Timings”, added Section 66.14.1 “FLEXCOM USART in Asynchronous Modes” and Section 66.14.3 “FLEXCOM TWI Timings”. |
| In Section 66.14.2 “FLEXCOM SPI Timings”, removed note below all tables from Table 66-52 “FLEXCOM0 in SPI Mode IOSET1 Timings” to Table 66-61 “FLEXCOM4 in SPI Mode IOSET3 Timings”. |
| Added Section 66.15 “USART in Asynchronous Modes”. |
| Section 66.16 “SPI Timings”: updated limitation on fSPCK in “Master Read Mode” and “Slave Write Mode”. Removed note below Table 66-63 “SPI0 IOSET1 Timings”, Table 66-64 “SPI0 IOSET2 Timings”, Table 66-65 “SPI1 IOSET1 Timings”, Table 66-66 “SPI1 IOSET2 Timings” and Table 66-67 “SPI1 IOSET3 Timings”. |
| Section 66.18 “QSPI Timings”: updated limitation on fQSCK in “Master Read Mode”. |
| Section 66.19.3 “LPDDR1-SDRAM”, Table 66-76 “System Clock Waveform Parameters”: updated min value of tDDRCK for VDDCORE[1.2V, 1.32V]. |
| Section 66.19.4 “LPDDR2/LPDDR3-SDRAM”, Table 66-77 “System Clock Waveform Parameters”: updated min value of tDDRCK for VDDCORE[1.2V, 1.32V]. |
| Section 66.19.5 “DDR3/DDR3L-SDRAM”: updated min values in Table 66-78 “System Clock Waveform Parameters”. |
| Section 66.23 “ISC Timings”: updated Figure 66-40 “ISC Timing Diagram” and Timings tables. |
| **Section 68. “Schematic Checklist”** |
| Table 68-1 “Power Supply Connections”: removed reference to VCCCORE. |
| Added Section 68.14.7 “Considerations for PTC Interface”. |
| **Section 70. “Ordering Information”** |
| Updated Table 70-1 “SAMA5D2 Ordering Information” with MRL C ordering codes. |
| **Section 71. “Errata”** |
| Added Section 71.1 “Errata - SAMA5D2 MRL C Parts”. |
| Section 71.2 “Errata - SAMA5D2 MRL B Parts”: added Section 71.1.1 “GMAC Timestamps and PTP packets”. |
| Section 71.3 “Errata - SAMA5D2 MRL A Parts”: added |
| - Section 71.1.1 “GMAC Timestamps and PTP packets” |
| - Section 71.3.4 “ROM Code: SPI Bootup Frequency”. |

## 71.4 Revision 11267E

| Changes |
|---|
| Deleted Section 61. “Security Module”. |

## 71.5 Revision 11267D

| Changes |
|---|
| Minor formatting and editorial changes throughout |
| **“Introduction”** |
| Updated listed DDR memories |
| **“Features”** |
| Frequency of digital fractional PLL for audio “11.289 MHz” corrected to “11.2896 MHz” |
| “Two 64-bit, 16-channel DMA controllers” changed to “51 DMA Channels including two 16-channel 64-bit Central DMA Controllers” |
| **Section 1. “Description”** |
| Updated description of Low-power modes |
| **Section 2. “Configuration Summary”** |
| “Class D amplifier” changed to “stereo Class D amplifier” |
| Updated text at end of section |
| **Section 3. “Block Diagram”** |
| Figure 3-1 “SAMA5D2 Series Block Diagram”: added ISC_MSK input; updated description of crystal oscillators; “PWMEXTRIG0-1” renumbered to “PWMEXTRG1–2” |
| Added note “See Section 35. “DMA Controller (XDMAC)” for peripheral connections to DMA.” |
| **Section 4. “Signal Description”** |
| Table 4-1 “Signal Description List”: NRST signal function “Microcontroller Reset” changed to “Microprocessor Reset”; “PWMEXTRG0-1” renumbered to “PWMEXTRG1–2”; “Self-refresh mode” changed to “Backup Self-refresh mode” in DDR_CKE comments |
| **Section 5. “Package and Pinout”** |
| Separated content into Section 5.1 “Packages” and Section 5.2 “Pinouts” |
| Table 5-2 “Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A)”: “ADVREFP” corrected to “ADVREF”; “PWMEXTRG0” and “PWMEXTRG1” renumbered to “PWMEXTRG1” and “PWMEXTRG2”; removed empty function cells for primary signals PA30, PA31, and PB0–PB7; removed “SEC, FILTER” from “Reset State” column header; added footnote on reset states |
| Added Table 5-3 “Pin Description (SAMA5D23 pins different from those in SAMA5D21/SAMA5D2)” and Table 5-4 “Pin Description (SAMA5D28B pins different from those in SAMA5D28A)” |
| **Section 6. “Power Considerations”** |
| Table 6-1 “SAMA5D2 Power Supplies”: updated rows VDDUTMIC, VDDHSIC and VDDOSC |
| Section 6.4.1 “VDDBU Power Architecture”: reworded second paragraph and deleted “typically less than 2 µA” |
| **Section 7. “Memories”** |
| Section 7.2.1 “External Bus Interface”: “The slew rates are determined by programming the SFR_EBICFG bit in SFR registers” changed to “The drive levels are configured with the DRIVEx field in the EBI Configuration Register (SFR_EBICFG)” |
| **Section 8. “Event System”** |
| Section 8-1 “Real-time Event Mapping List”: instance of “ADC_ADTRG” corrected to “ADTRG” |

.....continued

| Changes |
|---|
| **Section 9. “System Controller”** |
| Section 9.1 “Power-On Reset”: “dedicated to VDDBU, VDDIOP and VDDCORE” changed to “dedicated to monitoring VDDBU, VDDIOP and VDDCORE” |
| **Section 10. “Peripherals”** |
| Table 10-1 “Peripheral identifiers”: in ‘Instance Name’ column, renamed CAN0 and CAN1 to MCAN0 and MCAN1 |
| Section 10.4 “Peripheral Clock Types”: in SLOW_CLOCK description, “32768-Hz crystal oscillator or by the on-chip 32-kHz RC oscillator” changed to “32.768 kHz crystal oscillator or by the on-chip 64 kHz RC oscillator” |
| **Section 11. “Chip Identifier (CHIPID)”** |
| Updated Table 11-1 “SAMA5D2 Chip ID Registers” |
| **Section 13. “L2 Cache Controller (L2CC)”** |
| Table 13-2 “Register Mapping”: reset value 0x0000_0000 changed to 0x0000_0111 for L2CC_TRCR and L2CC_DRCR |
| **Section 14. “Debug and Test Features”** |
| Table 14-1 “Debug and Test Pin List”: NRST pin function “Microcontroller Reset” changed to “Microprocessor Reset” |
| **Section 15. “Standard Boot Strategies”** |
| “Boot Sequence Control Register (BSCR)” renamed to “Boot Sequence Controller Configuration Register” |
| Section 15.1 “Description”: “This microcontroller can be configured” changed to “This microprocessor can be configured” |
| Figure 15-10 “Galois Field Table Mapping”: modified Galois field table offsets |
| Section 15.4.2 “Boot Sequence Controller Configuration Register”: added address |
| Section 15.4.3 “Boot Configuration Word”: added reference to “Customer Fuse Matrix” |
| Added Section 15.4.6.5 “QSPI Flash Boot” |
| Table 15-3 “PIO Driven during Boot Program Execution”: NAND Flash PIO line PIOC17 changed to PIOB0 |
| **Section 18. “Special Function Registers (SFR)”** |
| Table 18-1 “Register Mapping”: removed EBI Configuration Register / SFR_EBICFG (offset 0x40 now reserved) |
| Section 18.3.1 “DDR Configuration Register”: added note |
| Removed section “EBI Configuration Register” |
| **Section 21. “Watchdog Timer (WDT)”** |
| Section 21.4 “Functional Description”: in eighth paragraph, “To prevent a software deadlock that continuously triggers the watchdog, the reload of the watchdog must occur...” changed to “The reload of the watchdog must occur...” |
| **Section 25. “Real-time Clock (RTC)”** |
| Reworked Section 25.5.6 “Updating Time/Calendar” |
| Reworked Figure 25-7 “Calibration Circuitry Waveforms” |
| AD index ‘7’ replaced with generic ‘n’ in Section 25.5.8 “Waveform Generation” |
| Updated Figure 25-8 “Waveform Generation for ADC Trigger Event” |
| Section 25.6.2 “RTC Mode Register”: |
| - updated descriptions of fields OUT0 and OUT1<br>- added fields TPERIOD and THIGH |
| **Section 27. “Low Power Asynchronous Receiver (RXLP)”** |

.....continued

| Changes |
|---|
| Pin/signal name “LPRXD” changed to “RXD” |
| **Section 29. “Clock Generator”** |
| Section 29.2 “Embedded Characteristics”: AUDIOPLLCK changed to AUDIOPLLCLK |
| Figure 29-1 “Clock Generator Block Diagram”: lines changed to arrows for OSCSEL to multiplexer, for MOSCSEL to multiplexer, and for PLLADIV2 to “PLLA and Divider” block |
| Figure 29-5 “Divider and PLLA Block Diagram”: added PLLADIV2 divider |
| Updated Section 29.8 “Audio PLL” |
| **Section 30. “Power Management Controller (PMC)”** |
| AUDIOPLLCK changed to AUDIOPLLCLK in Section 30.15 “Programmable Clock Controller” and Section 30.16 “Generic Clock Controller” |
| Figure 30-1 “General Clock Block Diagram”: added PLLA block; repositioned PLLACK signal; at bottom of diagram “PCKx” changed to “PCKx (to pads)” |
| Table 30-3 “Register Mapping”: PMC_AUDIO_PLL0 reset value ‘0x0000_0000’ changed to ‘0x0000_00D0’ |
| Section 30.22.11 “PMC Master Clock Register”: updated CSS field description |
| Section 30.22.13 “PMC Programmable Clock Register”: added addresses 0xF0014044 and 0xF0014048; updated CSS field description |
| Section 30.22.39 “PMC Audio PLL Control Register 0”: added fields DCO_FILTER (bits 29:28), DCO_GAIN (bits 27:24) and PLLFLT (bits 7:4) |
| Section 30.22.40 “PMC Audio PLL Control Register 1”: updated DIV field description |
| **Section 31. “Parallel Input/Output Controller (PIO)”** |
| Section 31.4.2 “External Interrupt Lines”: “are generally multiplexed” changed to “are multiplexed” |
| Section 31.5 “Functional Description”: removed entire section “Peripheral Muxing Example” |
| Table 31-4 “Register Mapping”: |
| - added reset value for PIO_CFGR, PIO_ODSR, PIO_IMR, S_PIO_CFGR, S_PIO_ODSR and S_PIO_IMR<br>- “PIO I/O Freeze Register” corrected to “PIO I/O Freeze Configuration Register” |
| - defined offset range 0x400–0x4FC as reserved |
| - reserved offset range 0x5E8–0x5F8 changed to 0x5E8–0x5FC |
| - “Secure PIO I/O Freeze Register” corrected to “Secure PIO I/O Freeze Configuration Register” |
| Removed duplicated or invalid addresses in Section 31.7.1 “PIO Mask Register”, Section 31.7.2 “PIO Configuration Register”, Section 31.7.3 “PIO Pin Data Status Register”, Section 31.7.4 “PIO Lock Status Register”, Section 31.7.5 “PIO Set Output Data Register”, and Section 31.7.6 “PIO Clear Output Data Register” |
| Section 31.7.7 “PIO Output Data Status Register”: removed duplicated or invalid addresses; access “Read-only or Read/Write” corrected to “Read/Write” |
| Removed duplicated or invalid addresses in Section 31.7.8 “PIO Interrupt Enable Register”, Section 31.7.9 “PIO Interrupt Disable Register”, Section 31.7.10 “PIO Interrupt Mask Register”, and Section 31.7.11 “PIO Interrupt Status Register” |
| Section 31.7.12 “PIO I/O Freeze Configuration Register”: corrected title (was “PIO Freeze Configuration Register”); removed duplicated or invalid addresses; access “Read/Write” corrected to “Write-only” |
| Removed duplicated or invalid addresses in Section 31.7.15 “Secure PIO Mask Register”, Section 31.7.16 “Secure PIO Configuration Register”, Section 31.7.17 “Secure PIO Pin Data Status Register”, Section 31.7.18 “Secure PIO Lock Status Register”, Section 31.7.19 “Secure PIO Set Output Data Register” and Section 31.7.20 “Secure PIO Clear Output Data Register” |
| Section 31.7.21 “Secure PIO Output Data Status Register”: removed duplicated or invalid addresses; access “Read-only or Read/Write” corrected to “Read/Write” |

.....continued

| Changes |
|---|
| Removed duplicated or invalid addresses in Section 31.7.22 “Secure PIO Interrupt Enable Register”, Section 31.7.23 “Secure PIO Interrupt Disable Register”, Section 31.7.24 “Secure PIO Interrupt Mask Register”, and Section 31.7.25 “Secure PIO Interrupt Status Register” |
| Section 31.7.29 “Secure PIO I/O Freeze Configuration Register”: corrected title (was “Secure PIO Freeze Configuration Register”); removed duplicated or invalid addresses; access “Read/Write” corrected to “Write-only” |
| Section 31.7.30 “Secure PIO Slow Clock Divider Debouncing Register”: added sentence about register write protection |
| **Section 32. “External Memories”** |
| Table 32-1 “DDR/LPDDR I/O Lines Description”: updated DDR_VREF function description |
| **Section 33. “Multiport DDR-SDRAM Controller (MPDDRC)”** |
| Section 33.4.1 “Low-power DDR1-SDRAM Initialization”: in first paragraph, removed content about configuring register SFR_DDRCFG |
| Section 33.6 “Software Interface/SDRAM Organization, Address Mapping”: modified description of Interleaved mode (“at each SDRAM end page” corrected to “at each DDRSDRAM end of page”) |
| Harmonized register naming throughout Section 33.7 “AHB Multiport DDR-SDRAM Controller (MPDDRC) User Interface” |
| Removed all MPDDRC DLL registers (offset range 0x100–0x158 now reserved) |
| Section 33.7.3 “MPDDRC Configuration Register”: modified description of DECOD bit value ‘1’ (“at each SDRAM end page” corrected to “at each DDR-SDRAM end of page”) |
| Section 33.7.12 “MPDDRC I/O Calibration Register”: updated RZQ values in RDIV field description |
| **Section 34. “Static Memory Controller (SMC)”** |
| Section 34.17.3 “NFC Initialization”: instances of “rbn” changed to “Ready/Busy” |
| Section 34.20.3 “NFC Status Register”: bit RB_EDGE3 (bit 27) replaced by RB_EDGE0 (bit 24); updated RB_RISE and RB_FALL bit descriptions |
| Bit RB_EDGE3 (bit 27) replaced by RB_EDGE0 (bit 24) in Section 34.20.4 “NFC Interrupt Enable Register”, Section 34.20.5 “NFC Interrupt Disable Register” and Section 34.20.6 “NFC Interrupt Mask Register” |
| Deleted invalid addresses in Section 34.20.30 “PMECC Error Location SIGMA0 Register” and Section 34.20.31 “PMECC Error Location SIGMAx Register” |
| Section 34.20.32 “PMECC Error Location x Register”: register index “x=0..23” corrected to “x=0..31” |
| Section 34.20.36 “Timings Register”: removed RBNSEL field |
| **Section 35. “DMA Controller (XDMAC)”** |
| Added XDMAC_CCx.CSIZE configuration to Table 35-2 “DMA Channels Definition (XDMAC0)” and Table 35-3 “DMA Channels Definition (XDMAC1)” |
| Table 35-5 “Register Mapping”: |
| - XDMAC_GCFG access Read-only corrected to Read/Write |
| - XDMAC_GWAC access Read-only corrected to Read/Write |
| Section 35.9.2 “XDMAC Global Configuration Register”: access Read-only corrected to Read/Write |
| Section 35.9.3 “XDMAC Global Weighted Arbiter Configuration Register”: access Read-only corrected to Read/Write |
| **Section 36. “LCD Controller (LCDC)”** |
| Updated ”Section 36.2 “Embedded Characteristics” |
| Updated Section 36.6.1.1 “Pixel Clock Period Configuration” |
| **Section 37. “Ethernet MAC (GMAC)”** |
| Table 37-1 “GMAC Connections in Different Modes”: added table Note on GTXCK |

| .....continued |
|---|
| **Changes** |
| Updated Section 37.5.3 “Interrupt Sources” |
| Section 37.7.1.2 “Receive Buffer List” and Section 37.7.1.3 “Transmit Buffer List”: added note at end of sections on queue pointer intilaization |
| Section 37.8.107 “GMAC Transmit Buffer Queue Base Address Register Priority Queue x” and Section 37.8.108 “GMAC Receive Buffer Queue Base Address Register Priority Queue x”: changed sentence on register initialization |
| **Section 39. “USB Host High Speed Port (UHPHS)”** |
| Section 39.2 “Embedded Characteristics”: “X Hosts (A and B) High Speed (EHCI)” corrected to “2 Hosts (A and B) High Speed (EHCI)” |
| Table 39-2 “Register Mapping”: inserted reserved offset 0x0C |
| Section 39.7.19 “EHCI: REG06 - AHB Error Status”: instances of “INSNREG[8:4]” changed to “INSNREG06[8:4]” |
| **Section 40. “Audio Class D Amplifier (CLASSD)”** |
| Section 40.2 “Embedded Characteristics”: DSP clock frequency “11.289 MHz” corrected to “11.2896 MHz” |
| Section 40.5.2 “Power Management”: field name “NOVRLAP” corrected to “NOVRVAL” |
| Figure 40-21 “Use Case 4B: Stereo Audio DAC With Passive Low Pass Filter and Single-ended Outputs”: changed title (was “Use Case 4B: Stereo Audio DAC With Passive Low Pass Filter and Differential Outputs”) |
| **Section 42. “Synchronous Serial Controller (SSC)”** |
| Figure 42-19 “Interrupt Block Diagram”: “RXSYNC” renamed to “RXSYN”; “TXSYNC” renamed to “TXSYN” |
| Section 42.8.10 “Register Write Protection”: in first sentence, “AIC behavior” corrected to “SSC behavior” |
| **Section 43. “Two-wire Interface (TWIHS)”** |
| Section 43.6.3.10 “SMBus Mode”: Deleted “A dedicated bus line, SMBALERT, allows a slave to get a master attention” from listed exceptions |
| Section 43.6.5.6 “SMBus Mode”: Deleted “A dedicated bus line, SMBALERT, allows a slave to get a master attention” from listed exceptions |
| Deleted note about debugger read access in Section 43.7.6 “TWIHS Status Register”, Section 43.7.13 “TWIHS Receive Holding Register” and Section 43.7.25 “TWIHS Write Protection Status Register” |
| **Section 44. “Flexible Serial Communication Controller (FLEXCOM)”** |
| Section 44.7.1.2 “Fractional Baud Rate in Asynchronous Mode”: in first paragraph, deleted sentence “This feature is only available when using USART Normal mode.” |
| Figure 44-8 “Preamble Patterns, Default Polarity Assumed”: instances of “8 bit width” changed to “8-bit” |
| Figure 44-11 “Asynchronous Start Detection”: added missing arrowheads |
| Section 44.7.3.11 “Receiver Timeout”: removed redundant paragraphs on STTTO and RETTO; reworded two bullets |
| Section 44.7.4 “ISO7816 Mode”: at end of second paragraph, “value 0x5 for protocol T = 1” changed to “value 0x6 for protocol T = 1” |
| Section 44.7.4.2 “Protocol T = 0”: reworded content under “Receive NACK Inhibit” |
| Section 44.7.7 “USART Comparison Function on Received Character”: modified information about the CMPMODE bit |
| Table 44-18 “Register Mapping”: added TWI SleepWalking Matching Register (FLEX_TWI_SWMR) |
| Section 44.10.41 “USART Write Protection Mode Register”: rephrased WPEN bit description |
| Corrected order of all sections from Section 44.10.66 “TWI Interrupt Enable Register” to Section 44.10.76 “TWI SleepWalking Matching Register” |
| Section 44.10.76 “TWI SleepWalking Matching Register”: added addresses |
| **Section 46. “Serial Peripheral Interface (SPI)”** |
| Figure 46-1 “Block Diagram”: added GCLK output from PMC to SPI |
| Modified transmission condition description in Section 46.7.3 “Master Mode Operations” |

| .....continued |
|---|
| **Changes** |
| Section 46.7.4 “SPI Slave Mode”: added sentence about NSS rising between characters |
| Section 46.7.5 “SPI Comparison Function on Received Character”: in seventh paragraph, added “if SleepWalking mode is disabled” to sentence “The comparison trigger event is...” |
| Updated Section 46.7.8 “Register Write Protection” |
| Section 46.8.2 “SPI Mode Register”: added bits BRSRCCLK (Bit Rate Source Clock) and LSBHALF (LSB Timing Selection); updated description of field DLYBCS |
| Section 46.8.12 “SPI Chip Select Register”: updated description of fields CSNAAT, SCBR, DLYBS and DLYBCT |
| **Section 47. “Quad Serial Peripheral Interface (QSPI)”** |
| Section 47.2 “Embedded Characteristics”: added bullet “Interface to Serial Flash Memories Operating in Single Data Rate or Double Data Rate Modes” |
| **Section 47. “Quad Serial Peripheral Interface (QSPI)” (cont’d)** |
| NSS renamed to QCS in Figure 47-2 “QSPI Transfer Format (QSPI_SCR.CPHA = 0, 8 bits per transfer)” and Figure 47-3 “QSPI Transfer Format (QSPI_SCR.CPHA = 1, 8 bits per transfer)” |
| Section 47.7.2 “QSPI Mode Register”: added note “This field is forced to LASTXFER when SMM is written to ‘1’ to CSMODE field description; modified equation in description of fields DLYBCT and DLYCS |
| Section 47.7.5 “QSPI Status Register”: updated descriptions of bits CSR and INSTRE |
| Section 47.7.9 “QSPI Serial Clock Register”: modified equation in description of fields SCBR and DLYBS |
| **Section 48. “Secure Digital Multimedia Card Controller (SDMMC)”** |
| Added Section 48.3 “Embedded Features for SDMMC0 and SDMMC1” |
| Figure 48-1 “Block Diagram”: added two notes |
| Table 48-3 “Register Mapping”: updated SDMMC_APSR reset value (SDMMC0 different from SDMMC1) |
| Section 48.13.18 “SDMMC Software Reset Register”: updated SWRSTCMD bit description |
| Section 48.13.58 “SDMMC Calibration Control Register”: in CNTVAL field description, “tSTARTUP = ...” corrected to “tSTARTUP = 2 µs” |
| **Section 49. “Image Sensor Controller (ISC)”** |
| Added Section 49.4 “Product Dependencies” |
| Table 49-18 “Register Mapping”: defined offset range 0x404–0x40C as reserved |
| **Section 50. “Controller Area Network (MCAN)”** |
| “GCLK3” changed to “GCLK” in Section 50.3 “Block Diagram” and Section 50.4.2 “Power Management” |
| Added Table 50-2 “Peripheral IDs” |
| Updated Section 50.5.1.3 “CAN FD Operation” |
| Section 50.5.1.4 “Transmitter Delay Compensation”: modified title (was “Transceiver Delay Compensation”); revised content |
| Section 50.5.1.5 “Restricted Operation Mode”: added ‘Note’ |
| Section 50.5.3 “Timeout Counter”: “baud rate” changed to “bit rate” in ‘Note’ |
| Section 50.5.4.1 “Acceptance Filtering”: “described in Section” corrected to “described in “Rx FIFO Overwrite Mode” |
| Updated Figure 50-5 “Standard Message ID Filter Path”and Figure 50-6 “Extended Message ID Filter Path” |
| Updated register names in Figure 50-7 “Rx FIFO Status” and Figure 50-8 “Rx FIFO Overflow Handling” |
| Section 50.5.7.2 “Rx Buffer and FIFO Element”: “R1 Bit 21 FDF: Extended Data Length” renamed to “R1 Bit 21 FDF: FD Format” |
| Section 50.5.7.4 “Tx Event FIFO Element”: “E1 Bit 21 FDF: Extended Data Length” renamed to “E1 Bit 21 FDF: FD Format” |

| .....continued |
|---|
| **Changes** |
| Table 50-14 “Register Mapping”: |
| - deleted row “0x00–0x04 / Reserved” |
| - “Fast Bit Timing and Prescaler Register” renamed to “Data Bit Timing and Prescaler Register” |
| #NAME? |
| Section 50.6.4 “MCAN Data Bit Timing and Prescaler Register”: changed name (was “MCAN Fast Bit Timing and Prescaler Register”); field FBRP replaced by field DBRP |
| Section 50.6.7 “MCAN CC Control Register”: updated descriptions of fields FDOE, BRSE, PXHD and EFB; removed NISO bit |
| Section 50.6.8 “MCAN Nominal Bit Timing and Prescaler Register”: “NBRP: Nominal Baud Rate Prescaler” changed to “NBRP: Nominal Bit Rate Prescaler” |
| Section 50.6.9 “MCAN Timestamp Counter Configuration Register”: updated TSS field description |
| Section 50.6.10 “MCAN Timestamp Counter Value Register”: updated TSC field description |
| Section 50. “Controller Area Network (MCAN)” (cont’d) |
| Section 50.6.20 “MCAN Global Filter Configuration”: added details on register description; updated ANFE and ANFS field descriptions. |
| Added details on register description in Section 50.6.21 “MCAN Standard ID Filter Configuration” and Section 50.6.22 “MCAN Extended ID Filter Configuration” |
| Section 50.6.24 “MCAN High Priority Message Status”: updated description of MSI field value ‘1’ |
| **Section 51. “Timer Counter (TC)”** |
| Replaced TIOA, TIOB, TCLK with TIOAx, TIOBx, TCLKx |
| Table 51-1 “Timer Counter Clock Assignment”: updated definitions |
| Section 51.6.3 “Clock Selection”: updated bullet “Internal clock signals”, updated notes 1 and 2 |
| Section 51.6.9 “Transfer with DMAC in Capture Mode”: updated title (added “in Capture Mode”) |
| Updated Figure 51-5 “Example of Transfer with DMAC in Capture Mode” |
| Section 51.6.16.4 “Position and Rotation Measurement”: updated text in first paragraph |
| Added Section 51.6.16.6 “Detecting a Missing Index Pulse” |
| Updated TCCLKS field description in Section 51.7.2 “TC Channel Mode Register: Capture Mode” and Section 51.7.3 “TC Channel Mode Register: Waveform Mode” |
| **Section 53. “Pulse Width Modulation Controller (PWM)”** |
| Throughout, “PWMTRG” and “EXTTRG” renamed to “PWMEXTRG” |
| Table 53-2 “I/O Lines”: “PWMEXTRG0” and “PWMEXTRG1” renumbered to “PWMEXTRG1” and “PWMEXTRG2” |
| Updated Section “Recoverable Fault” |
| Updated Figure 53-1 “Pulse Width Modulation Controller Block Diagram” and added note below figure |
| Updated Figure 53-16 “Fault Protection” |
| **Section 54. “Secure Fuse Controller (SFC)”** |
| Table 54-1 “Register Mapping”: removed reset value from SFC_IER and SFC_IDR (both registers are write-only) |
| **Section 55. “Integrity Check Monitor (ICM)”** |
| Table 55-8 “Register Mapping”: ICM_SR access “Write-only” corrected to “Read-only” |
| **Section 57. “Advanced Encryption Standard (AES)”** |
| Table 57-5 “Register Mapping”: AES_ALPHAR[0..3] access “Write” corrected to “Write-only” |

| .....continued |
|---|
| **Changes** |
| Section 57.5.20 “AES Alpha Word Register x”: access “Write” corrected to “Write-only” |
| **Section 59. “Triple Data Encryption Standard (TDES)”** |
| Section 59.4.1 “Operating Modes”: deleted sentence “The OFB and CFB modes of operation are only available if 2-key mode is selected (KEYMOD = 1 in TDES_MR).” |
| Section 59.4.3 “Last Output Data Mode”: deleted sentence “No more Output Data Register reads are necessary between consecutive encryptions/decryptions (see Section 59.4.3 “Last Output Data Mode”).” |
| Section 59.5.2 “TDES Mode Register”: in OPMOD field description, deleted sentence “The OFB and CFB modes of operation are only available if 2-key mode is selected (KEYMOD = 1).” |
| **Section 61. “Security Module”** |
| Updated Figure 61-2 “Security Module Internal Memory Map” |
| **Section 62. “Analog-to-Digital Converter (ADC)”** |
| Section 62.1 “Description”: |
| - deleted sentence “A digital error correction circuit based on the multi-bit redundant signed digit (RSD) algorithm is implemented to reduce INL and DNL errors.” |
| - deleted sentence “Finally, the user can configure ADC timings, such as startup time and tracking time.” |
| Updated Section 62.2 “Embedded Characteristics” |
| Updated Figure 62-1 “Analog-to-Digital Converter Block Diagram” |
| Revised Section 62.5 “Product Dependencies” |
| Revised Section 62.6.1 “Analog-to-Digital Conversion” |
| Updated Section 62.6.3 “ADC Reference Voltage” and Section 62.6.4 “Conversion Resolution” |
| Updated Section 62.6.7 “Conversion Triggers” |
| Section 62.6.9 “Comparison Window”: in fourth paragraph, instance of “ADC_SR” corrected to “ADC_ISR” |
| Section 62.6.10 “Differential and Single-ended Input Modes”: changed title (was “Differential Inputs”) and revised content |
| Updated Section 62.6.11 “ADC Timings”, Section 62.6.12 “Last Channel Specific Measurement Trigger”, Section 62.6.13 “Enhanced Resolution Mode and Digital Averaging Function” and Section 62.6.14 “Automatic Error Correction” |
| Instances of GND renamed to GNDANA in Figure 62-15 “Touchscreen Switches Implementation”, Figure 62-18 “Touchscreen Switches Implementation” and Figure 62-20 “Touchscreen Pen Detect” |
| Updated Section 62.6.16 “Asynchronous and Partial Wakeup (SleepWalking)” |
| Section 62.6.17.1 “Classic ADC Channels Only (Touchscreen Disabled)”: changed title (was “Classical ADC Channels Only”) |
| Section 62.6.19 “Register Write Protection”: updated list of protectable registers |
| Table 62-8 “Register Mapping”: |

| .....continued |
|---|
| **Changes** |
| - defined 0x48 as reserved<br>- added row 0x4C / Channel Offset Register / ADC_COR<br>- added offset 0x7C for ADC_CDR11<br>- defined offset range 0x80–0x90 as reserved<br>- added row 0x94 / Analog Control Register / ADC_ACR<br>- defined offset range 0xC4–0xD0 as reserved<br>- added row 0xD4 / Correction Values Register / ADC_CVR<br>- added row 0xD8 / Channel Error Correction Register / ADC_CECR<br>- added row 0xDC / Touchscreen Correction Values Register / ADC_TSCVR<br>- defined offset 0xE0 as reserved |
| Section 62.7.2 “ADC Mode Register”: updated TRACKIM field description |
| Added LCCHG (Last Channel Change) bit in Section 62.7.9 “ADC Interrupt Enable Register”, Section 62.7.10 “ADC Interrupt Disable Register”, Section 62.7.11 “ADC Interrupt Mask Register” and Section 62.7.12 “ADC Interrupt Status Register” |
| Section 62.7.13 “ADC Last Channel Trigger Mode Register”: updated CMPMOD field description |
| Section 62.7.16 “ADC Extended Mode Register”: updated CMPMODE field description; added descriptions for fields OSR ASTE |
| Section 62.7.18 “ADC Channel Offset Register”: added address; removed bits OFF11:OFF0 from bitmap; modified DIFFx field description |
| Section 62.7.20 “ADC Analog Control Register”: added address; added IBCTL field |
| Section 62.7.25 “ADC Trigger Register”: added sentence about write protection |
| Removed Section “Correction Select Register” |
| Added sentence about write protection in Section 62.7.26 “ADC Correction Values Register” and Section 62.7.27 “ADC Channel Error Correction Register” |
| Added Section 62.7.28 “ADC Touchscreen Correction Values Register” |
| **Section 63. “Electrical Characteristics”** |
| “ADVREFP” corrected to “ADVREF” |
| Section 63.2 “DC Characteristics”: in first sentence, “TA = -40°C to +85°C” changed to “TA = -40°C to +105°C” |
| Added Table 63-2 “Recommended Thermal Operating Conditions” |
| Updated Section 63.4 “Active Mode” |
| Table 63-8 “Low-power Mode Configuration Summary”: updated values for ‘Consumption’ and ‘Wakeup Time’ |
| Updated Section 63.5.6 “Low-power Consumption Versus Modes” |
| Table 63-9 “Typical Power Consumption in Idle Mode: AMP2”: updated consumption values |
| Table 63-10 “VDDCORE Power Consumption in Ultra Low-power Mode: AMP2”: updated consumption values; updated wakeup time for ULP1 Fast Wakeup mode |
| Table 63-11 “Typical Power Consumption for Backup Mode”: updated consumption values |
| Updated Table 63-12 “Processor Clock Waveform Parameters” and Table 63-13 “Master Clock Waveform Parameters” |
| Updated Section 63.7.1 “Main Oscillator Characteristics” |
| Table 63-17 “12 MHz RC Oscillator Characteristics”: updated startup time values |
| Updated Section 63.7.3 “32.768 kHz Crystal Oscillator Characteristics” |
| Updated Table 63-23 “Audio PLL Characteristics” |

| .....continued |
|---|
| **Changes** |
| Section 63.10 “ADC Characteristics”: deleted sentence “The VREFN pin must be connected to ground.” |
| Table 63-25 “Power Supply Characteristics”: updated Analog Current Consumption value for Fast Wakeup mode |
| Table 63-26 “ADVREF Electrical Characteristics”: “VREFP” corrected to “ADVREF”; updated Current value |
| Section 63.10.4.1 “Differential Mode (12-bit mode)” and Section 63.10.4.2 “Single-ended Mode (12-bit mode)”: in equation, “VVREFP” corrected to “VADVREF” |
| Section 63.10.4.4 “Gain and Offset Errors”: “VVREFP” corrected to “VADVREF” |
| Table 63-27 “ADC Timing Characteristics”: updated footnote |
| Added Table 63-32 “ADC Analog Input Characteristics” |
| Table 63-37 “VDDCORE Power-On Reset Characteristics”: updated Hysteresis Voltage values |
| Section 63.14.1 “Maximum SPI Frequency”: updated values in “Master Read Mode” and “Slave Write Mode” |
| Revised Section 63.18 “MPDDRC Timings” |
| Corrected CKI values in Figure 63-33 “SSC Transmitter, TK and TF in Input”, Figure 63-35 “SSC Receiver, RK in Input and RF in Output”, Figure 63-36 “SSC Receiver, RK and RF in Output” and Figure 63-38 “Minimum and Maximum Access Time of Output Signals” |
| **Section 65. “Schematic Checklist”** |
| Figure 65-1 “1.2V, 1.35V/1.5V, 2V, 2.5V, 3.3V Power Supplies Schematics(1)”: GNDHSIC changed to GNDUTMIC |
| Table 65-1 “Power Supply Connections”: updated GNDUTMIC row; removed GNDHSIC row; in second footnote, “microcontroller” changed to “microprocessor” |
| Table 65-2 “Clock, Oscillator and PLL Connections”: “(internal 32-kHz RC oscillator) changed to “(internal 64 kHz RC oscillator)” |
| Section 65.5.1 “How to Define the Oscillator Load Capacitance”: instances of “32-KHz Oscillator” changed to “32.768 kHz Oscillator” |
| Added Section 65.14.6 “QSPI Pull-up Resistors” |
| Updated **Section 67. “Ordering Information”** |
| **Section 68. “Errata”** |
| Updated and reorganized content (errata now collected in Section 68.1 “Errata - SAMA5D2 MRL-B Parts” and Section 68.2 “Errata - SAMA5D2 MRL-A Parts”) |

## 71.6 Revision 11267C

| Changes |
|---|
| Changed datasheet status from ‘Preliminary’ to ‘Complete’. |
| Added “Introduction” and transferred Description to Section 1. |
| **Section 2. “Configuration Summary”** |
| Added device compatibility information |
| **Section 4. “Signal Description”** |
| Table 4-1 “Signal Description List”: modified rows PIOBU 0-7 and DDR_RESETN |
| **Section 6. “Power Considerations”** |
| Added Section 6.4.1 “VDDBU Power Architecture” |
| Updated Section 6.2 “Power-Up Considerations” and Section 6.3 “Power-Down Considerations” |
| **Section 7. “Memories”** |

| .....continued |
|---|
| **Changes** |
| Updated Section 7.1.2 “Internal ROM” |
| **Section 10. “Peripherals”** |
| Updated Table 10-1 “Peripheral identifiers” and Section 10.4 “Peripheral Clock Types” |
| **Section 16. “AXI Matrix (AXIMX)”** |
| Table 16-1 “Register Mapping”: removed 0x00000000 reset value from all rows |
| **Section 17. “Matrix (H64MX/H32MX)”** |
| Section 17.2 “Embedded Characteristics”: removed "Master number forwarding to slaves" characteristic |
| Updated Table 17-1 “List of H64MX Masters”, Table 17-2 “List of H64MX Slaves”, Table 17-4 “List of H32MX Masters”, Table 17-5 “List of H32MX Slaves” |
| Table 17-3 “Master to Slave Access on H64MX”: updated ‘SDMMC0-SDMMC1’ row |
| Table 17-6 “Master to Slave Access on H32MX”: updated ‘Slave 5’ rows |
| Section 17.12.2 “Security of APB Slaves”: added introduction and bulleted list introduced by “As a general rule” |
| Added Section 17.12.3 “Security Types of AHB Master Peripherals” and Section 17.12.4 “Security Types of AHB Slave Peripherals” |
| Section 17-9 “Peripheral Identifiers”: corrected some security type names |
| Section 17.13 “AHB Matrix (MATRIX) User Interface”: added introduction and modified reset value of Updated Security Areas Split Slave x Registers in Table 17-10 “Register Mapping” |
| **Section 21. “Watchdog Timer (WDT)”** |
| Replaced “Idle mode” with “Sleep mode (Idle mode)” in Section 21.1 “Description” and with “Sleep mode” in Section 21.4 “Functional Description” |
| **Section 22. “Reset Controller (RSTC)”** |
| Renamed 'proc_nreset' to 'Processor Reset', 'periph_nreset' to 'Peripheral Reset', 'backup_neset' to 'Backup Reset', 'rstc_irq' to 'Reset Controller Interrupt', 'wd_fault' to 'Watchdog Fault', ‘user_reset’ to User Reset. |
| Updated text and figures to show that Processor Reset and Peripheral Reset signals are merged. |
| **Section 23. “Shutdown Controller (SHDWC)”** |
| Updated Figure 23-1 “Shutdown Controller Block Diagram” and Table 23-1 “I/O Lines Description” |
| Section 23.7.3 “Shutdown Status Register”: corrected register table (added WKUPIS9) |
| Section 23.7.4 “Shutdown Wake-up Inputs Register”: corrected register table (added WKUPT9 and WKUPEN9) |
| **Section 25. “Real-time Clock (RTC)”** |
| Removed RTC Milliseconds Register (RTC_MSR) and all related information in Section 25.1 “Description”, Section 25.2 “Embedded Characteristics”, Section 25.5 “Functional Description” and Section 25.6 “Real-time Clock (RTC) User Interface”. |
| Table 25-1 “Register Mapping”: modified RTC_CALR reset value |
| Section 25.6.1 “RTC Control Register”: updated CALEVSEL field description |
| Updated Section 25.6.22 “RTC TimeStamp Source Register” |
| **Section 29. “Clock Generator”** |
| Section 29.2 “Embedded Characteristics”: replaced "400 to 1000 MHz programmable PLL" with "600 to 1200 MHz programmable PLL" and replaced “HCLOCK” with “HCLOCK_LS/HS” and “PCLOCK” with “PCLOCK_LS/HS” |
| Section 29.4 “Slow Clock”: removed “This allows the slow clock to be valid in a short time (about 100 µs)” |
| Section 29.8 “Audio PLL”: updated all equations and added “in the 700 MHz range” after “The PLL core operates at 700 MHz (AUDIOCORECLOCK)” |

| .....continued |
|---|
| **Changes** |
| Updated Figure 29-3. Main Clock Block Diagram and Figure 29-4. Main Clock Source Selection |
| **Section 30. “Power Management Controller (PMC)”** |
| Updated Section 30.6 “Matrix Clock Controller” |
| Updated Section 30-1 “General Clock Block Diagram” |
| Section 30.19 “Programming Sequence”, sub-section “Selecting Master Clock and Processor Clock”: updated sequence following "If a new value for CSS field corresponds to PLL Clock" |
| Section 30.22.11 “PMC Master Clock Register”: updated H32MXDIV field description |
| **Section 33. “Multi-port DDR-SDRAM Controller (MPDDRC)”** |
| Section 33-2 “Single Write Access, Row Closed, DDR-SDRAM Devices” to Section 33-8 “SINGLE Write Access Followed by a Read Access, DDR2-SDRAM Devices”: replaced “D[15:0]” with “DATA” |
| Updated Section 33.7.9 “MPDDRC Low-power DDR2 Low-power DDR3 Low-power Register” |
| Section 33.7.10 “MPDDRC Low-power DDR2 Low-power DDR3 and DDR3 Calibration and MR4 Register”: updated MR4_READ field description |
| **Section 34. “Static Memory Controller (SMC)”** |
| Removed NFCCMD field and modified Section 34.17.2.1 “Building NFC Address Command Example” and Section 34.17.2.2 “NFC Address Command” accordingly |
| Table 34-20 “Register Mapping”: corrected offset values of PMECC Error Location 31 Register and of subsequent reserved range; removed reset value from HSMC_CTRL (register is write-only) |
| **Section 35. “DMA Controller (XDMAC)”** |
| Section 35.5.4.1 “Single Block With Single Microblock Transfer”: added text on memory-to-memory transfer |
| Section 35.8 “XDMAC Software Requirements”: added bullet on memory-to-memory transfer |
| Table 35-5 “Register Mapping”: corrected access of XDMAC_GTYPE, XDMAC_GWAC, XDMAC_CIM |
| Section 35.9.6 “XDMAC Global Interrupt Mask Register”: corrected access to Read-only |
| Section 35.9.28 “XDMAC Channel x [x = 0..15] Configuration Register”: corrected INITD and PERID field descriptions |
| **Section 36. “LCD Controller (LCDC)”** |
| Modified width of fields in Section 36.7.2 “LCD Controller Configuration Register 1” and Section 36.7.3 “LCD Controller Configuration Register 2” |
| **Section 40. “Audio Class D Amplifier (CLASSD)”** |
| Replaced ‘audio clock’ with ‘generic clock’ and ‘ACLK’ with ‘GCLK’ throughout the section |
| **Section 41. “Inter-IC Sound Controller (I2SC)”** |
| Section 41.6.3 “Master, Controller and Slave Modes”: removed text fragment: ‘in order to avoid unwanted glitches on the I2SWS and I2SCK pins.’ |
| Section 41.8.2 “Inter-IC Sound Controller Mode Register”: removed text fragment: ‘in order to avoid unexpected behavior on the I2SWS, I2SCK and I2SDO outputs.’ and added note (2) below IMCKDIV field description. |
| **Section 44. “Flexible Serial Communication Controller (FLEXCOM)”** |
| Restored all references to ISO7816 specification |
| Updated Figure 44-3 “Fractional Baud Rate Generator” |
| Added Figure 44-27 “RTS line software control when FLEX_US_MR.USART_MODE = 2” |
| Section 44.10.6 “USART Mode Register”: updated USART_MODE field description (SPI_MASTER item) |

| .....continued |
|---|
| **Changes** |
| Section 44.10.44 “SPI Mode Register”: added LBHPC bit |
| **Section 45. “Universal Asynchronous Receiver Transmitter (UART)”** |
| Section 45.6.9 “UART Baud Rate Generator Register”: in CD field description, corrected equation after “If BRSRCCK = 1” |
| **Section 47. “Quad SPI Interface (QSPI)”** |
| Section 47.7.5 “QSPI Status Register”: updated RDRF, TDRE, TXEMPTY, and OVRES field descriptions |
| **Section 48. “Secure Digital Multimedia Card (SDMMC)”** |
| Section 48.12.41 “SDMMC Preset Value Register”: updated CLKGSEL field description |
| **Section 49. “Image Sensor Controller (ISC)”** |
| Section 49.1 “Description”: removed "serial csi-2 based CMOS/CCD sensor" (not supported). |
| **Section 50. “Controller Area Network (MCAN)”** |
| Changed MCAN interrupt line names to MCAN_INT0 and MCAN_INT1 thoughout the section |
| Section 50.6.7 “MCAN CC Control Register”: added bit NISO |
| **Section 51. “Timer Counter (TC)”** |
| Reformatted and renamed Table 51-2 “Channel Signal Description” |
| Section 51.6.3 “Clock Selection”: updated notes (1) and (2) |
| **Section 52. “Pulse Density Modulation Interface Controller (PDMIC)”** |
| Replaced all instances of “PCK” with “GCLK” |
| Section 52.2 “Embedded Characteristics”: removed ‘Multiplexed PDM Input Support’ characteristic |
| Updated Section 52.5.2 “Power Management” and Section 52.6.2.1 “Description” |
| Section 52.6.2.6 “Gain and Offset Compensation”: updated dgain bullet |
| Section 52.7.3 “PDMIC Converted Data Register”: updated DATA field description |
| Section 52.7.8 “PDMIC DSP Configuration Register 0”: updated OSR field description |
| **Section 61. “Security Module”** |
| Section 61.5.5 “SECUMOD Status Clear Register”: removed MCKM field description |
| Section 61.5.18 “SECUMOD Wake Up Register”: removed TPML field description |
| **Section 62. “Analog-to-Digital Converter (ADC)”** |
| Section 62.7.2 “ADC Mode Register”: updated TRACKTIM and TRANSFER field descriptions. |
| **Section 63. “Electrical Characteristics”** |
| Updated tables from Table 63-2 “DC Characteristics” to Table 63-33 “Analog Comparator Characteristics” |
| Updated Figure 63-2 “Main Oscillator Schematics” |
| Corrected Gain Error formula under Figure 63-5 “Gain and Offset Errors in Single-ended Mode” |
| Removed Figure 63-4 “Single-ended Mode ADC” and Figure 63-5 “Differential Mode ADC” |
| Updated wake-up pin numbers in Section 63.5.1 “Backup Mode” and Section 63.5.3.2 “ULP1 Mode” |
| Updated Section 63.5.4 “Idle Mode” and Section 63.23 “SDMMC Timings” |
| Section 63.14.3 “Timing Extraction”: added introduction and Figure 63-11 “MISO Capture in Master Mode” |
| **Section 65. “Schematic Checklist”** |

| .....continued |
|---|
| **Changes** |
| Removed Table 65-12. “EBI Pins and NAND Flash Device Connections” and Table 65-13. “DDR2 I/O Lines Usage vs Operating Modes” |
| Reorganized **Section 67. “Ordering Information”** |
| **Updated Section 68. “Errata”** |

## 71.7 Revision 11267B

| **Changes** |
|---|
| “Features” |
| Updated Security features |
| **Section 2. “Block Diagram”** |
| Updated Figure 2-1 “SAMA5D2 Series Block Diagram”. |
| **Section 4. “Package and Pinout”** |
| Updated Table 4-2 “Pin Description” |
| Removed Section 4.2 “Input/Output Description” and Section 4-3 “SAMA5D2 I/O Type Description” |
| **Section 5. “Power Considerations”** |
| Updated Table 5-1 “SAMA5D2 Power Supplies” |
| Updated Figure 5-1 “Recommended Power-Up Sequence”, Figure 5-2 “Recommended Power-Down Sequence”, Figure 5-3 “Recommended Backup Mode Entry”, Figure 5-4 “Recommended Power Supply Sequencing at Wake-Up” |
| **Section 7. “Event System”** |
| Updated Table 7-1 “Real-time Event Mapping List” |
| **Section 14. “Standard Boot Strategies”** |
| Replaced all instances of "GPBR" with "BUREG". |
| **Section 17. “Special Function Registers (SFR)”** |
| Updated Section 17.3.15 “I2S Register” |
| **Section 19. “Advanced Interrupt Controller (AIC)”** |
| Removed Sections “Interrupt Vectoring” and “Fast Interrupt Vectoring” |
| Updated Section 19.8.3.3 “Interrupt Handlers”and Section 19.8.4.3 “Fast Interrupt Handlers” |
| **Section 29. “Power Management Controller (PMC)”** |
| Replaced “generated clock” with “generic clock”, and “GCK” with “GCLK” |
| Updated Section 29.22.8 “PMC Clock Generator Main Oscillator Register” |
| **Section 30. “Parallel Input/Output Controller (PIO)”** |
| Removed all references to programmable I/O delay |
| Added **Section 31. “External Memories”** |
| **Section 32. “Multi-port DDR-SDRAM Controller (MPDDRC)”** |
| Section 32.4.3 “Low-power DDR2-SDRAM Initialization”: added Step 14., Step 15. and Step 21. |
| Section 32.4.5 “Low-power DDR3-SDRAM Initialization”: added Step 14., Step 15. and Step 21. |
| Section 32.7.8 “MPDDRC Memory Device Register”: updated DBW field description; corrected location of fields RL3 and WL |

| .....continued |
|---|
| **Changes** |
| **Section 36. “Ethernet MAC (GMAC)”** |
| Updated Section 36.1 “Description” |
| Section 36.5.2 “Power Management”: deleted reference to PMC_PCER |
| Section 36.5.3 “Interrupt Sources”: deleted reference to ‘Advanced Interrupt Controller’. Replaced by ‘Interrupt Controller’. |
| Section 36.6.14 “IEEE 1588 Support”: deleted reference to GMAC_TSSx. Removed reference to ‘output pins’ in 2nd paragraph. |
| Section 36.6.15 “Time Stamp Unit”: added information on GTSUCOMP signal in last paragraph |
| **Section 39. “Audio Class D Amplifier (CLASSD)”** |
| Updated Figure 39-1. CLASSD Block Diagram |
| **Section 40. “Inter-IC Sound Controller (I2SC)”** |
| Replaced all instances of “PCKx” with “GCLK” |
| Removed all references to Time Division Multiplexed (TDM) format (not supported) |
| Section 40.1 “Description”: replaced “The I2SC can use either a single DMA Controller channel for both audio channels or one DMA Controller channel per audio channel.” with “The I2SC uses a single DMA Controller channel for both audio channels.”, and updated Section 40.2 “Embedded Characteristics” and Section 40.6.8 “DMA Controller Operation” accordingly |
| Section 40.8.2 “Inter-IC Sound Controller Mode Register”: removed fields RXDMA and TXDMA |
| **Section 43. “Flexible Serial Communication Controller (FLEXCOM)”** |
| Added SPI mode in UART/USART |
| Replaced all instances of ‘PCK’ with ‘GCLK’ |
| Replaced all instances of ‘DMAC/PDC’ with ‘DMAC’ |
| Removed SleepWalking characteristic from UART/USART mode |
| Removed all references to ISO7816 specification |
| Section 43.10.6 “USART Mode Register” updated USCLKS field description |
| Section 43.10.42 “SPI Mode Register”: updated BRSRCCLK and DLYBCS field descriptions |
| Section 43.10.52 “SPI Chip Select Register”: updated CSNAAT, SCBR, DLYBS and DLYBCT field descriptions |
| Section 43.10.62 “TWI Clock Waveform Generator Register”: updated BRSRCCLK and CKSRC field descriptions |
| Updated Figure 43-1 “FLEXCOM Block Diagram” and Figure 43-63 “Master Mode Block Diagram” |
| **Section 42. “Two-wire Interface (TWIHS)”** |
| Replaced all instances of “PMC_PCK” with “GCLK” |
| **Section 44. “Universal Asynchronous Receiver Transmitter (UART)”** |
| Replaced "Processor-Independent Source Clock" with "Processor-Independent Generic Source Clock" and "PCK" with "GCLK" |
| **Section 47. “Secure Digital Multimedia Card (SDMMC)”** |
| Updated revision of supported e.MMC specification (from V4.41 to V4.51) |
| **Section 51. “Pulse Density Modulation Interface Controller (PDMIC)”** |
| Removed all references to PDC |
| Removed Section 1.6.4 “Buffer Structure” |
| **Section 53. “Secure Fuse Controller (SFC)”** |
| Removed all references to lock fuse (not supported) |

| .....continued |
|---|
| **Changes** |
| Section 53.4.5.3 “Fuse Masking”: corrected data register names |
| Section 53.5.2 “SFC Mode Register”: updated MSK field description |
| Table 53-1 “Register Mapping”: modified SFC_IER and SFC_IDR access type from “Read/Write” to “Write-only” |
| **Section 56. “Advanced Encryption Standard (AES)”** |
| Updated Figure 56-12 “Generation of an ESP IPSec Frame without ESN” and Figure 56-13 “Generation of an ESP IPSec Frame with ESN” |
| Added **Section 60. “Security Module”** |
| **Section 61. “Analog-to-Digital Converter (ADC)”** |
| Updated enhanced resolution value from 12 bits to 14 bits |
| Renamed “Hold time” to “Transfer time” |
| Replaced all instances of “PMC PCK” with “GCLK” |
| Added Section 61.6.6 “Conversion Results Format”, Section 61.7.13 “ADC Last Channel Trigger Mode Register”, Section 61.7.14 “ADC Last Channel Compare Window Register” |
| Section 61.2 “Embedded Characteristics”: corrected conversion rate |
| Section 61.6.9 “Comparison Window”: added paragraph about bit SIGNMODE |
| Section 61.6.14 “Automatic Error Correction”: replaced “GAIN_ERROR_SIZE-1” with appropriate value; replaced “Gs-1” with “Gs” in formulas |
| Section 61.6.14 “Automatic Error Correction”, Section 61.7.27 “Correction Values Register”: replaced “GAIN_ERROR_SIZE-1” and “OFFSET_ERROR_SIZE-1” with appropriate values |
| Section 61.7.2 “ADC Mode Register”: updated TRGSEL and TRACKTIM field descriptions |
| Updated Section 61.7.8 “ADC Last Converted Data Register” |
| Section 61.7.16 “ADC Extended Mode Register”: added bit SIGNMODE |
| Updated Section 61.7.18 “ADC Channel Offset Register” |
| Updated Figure 61-1 “Analog-to-Digital Converter Block Diagram” and Figure 61-7 “Analog Full Scale Ranges in Single-Ended/Differential Applications” |
| Updated Table 61-5 “Oversampling Digital Output Range Values” |
| **Section 62. “Electrical Characteristics”** |
| Added: |
| - Section 62.11 “Analog Comparator Characteristics” |
| - Section 62.14.1 “Maximum SPI Frequency” |
| - Section 62.16.1 “Maximum QSPI Frequency” |
| - Table 62-3 “I/O Switching Frequency” |
| - Table 62-4 “QSPI I/O Switching Frequency” |
| - Table 62-21 “PLL UTMI Characteristics” |
| - Table 62-22 “PLLAUDIO Characteristics” |
| Updated: |
| - Table 62-1 “Absolute Maximum Ratings*” |
| - Table 62-2 “DC Characteristics” |
| - Table 62-6 “Typical Peripheral Power Consumption by Peripheral in Active Mode” to Table 62-10 “Typical Power Consumption for Backup Mode” |

| .....continued |
|---|
| **Changes** |
| - Table 62-13 “8 to 24 MHz Crystal Oscillator Characteristics” |
| - Table 62-16 “12-MHz RC Oscillator Characteristics” to Table 62-21 “PLL UTMI Characteristics” |
| - Table 62-34 “VDDBU Power-On Reset Characteristics” to Table 62-36 “VDDANA Power-On Reset Characteristics” |
| Reworked Section 62.9 “USB HS Characteristics” |
| **Section 64. “Schematic Checklist”** |
| Updated: |
| - Section 64.16.3 “DDR Layout and Design Considerations” |
| - Figure 64-1 “1.2V, 1.35V/1.5V, 2V, 2.5V, 3.3V Power Supplies Schematics(1)” |
| - Table 64-1 “Power Supply Connections” |

## 71.8 Revision 11267A

| Issue Date | Changes |
|---|---|
| 10-Sep-15 | Preliminary Datasheet - First issue |
| 25-Feb-15 | Advance Information Datasheet. |

# Table of Contents


Introduction....1

Features....1

1. Description....5

2. Configuration Summary....6

3. Block Diagram....7

4. Signal Description....8

5. Safety and Security Features....15

5.1. Design for Safety and IEC60730 Class B Certification....15
5.2. Design for Security....15
5.3. Safety and IEC 60730 Features....15
5.4. Security Features....17

6. Package and Pinout....20

6.1. Packages....20
6.2. Pinouts....20

7. Power Considerations....51

7.1. Power Supplies....51
7.2. Power-up Considerations....51
7.3. Power-down Considerations....53
7.4. Power Supply Sequencing at Backup Mode Entry and Exit....54

8. Memories....57

8.1. Embedded Memories....58
8.2. External Memory....58

9. Event System....62

9.1. Real-time Event List....62
9.2. Real-time Event Mapping....62

10. System Controller....64

10.1. Power-On Reset....66

11. Peripherals....67

11.1. Peripheral Mapping....67
11.2. Peripheral Identifiers....67
11.3. Peripheral Signal Multiplexing on I/O Lines....67
11.4. Peripheral Clock Types....67

12. Chip Identifier (CHIPID)....69
12.1. Description....69
12.2. Embedded Characteristics....69
12.3. Register Summary....71

13. Cortex-A5 Processor (ARM)....76
13.1. Reference Documents....76
13.2. Description....76
13.3. Embedded Characteristics....77
13.4. Block Diagram....77
13.5. Programmer Model....78
13.6. Memory Management Unit (MMU)....86

14. L2 Cache Controller (L2CC)....92
14.1. Description....92
14.2. Embedded Characteristics....92
14.3. Product Dependencies....92
14.4. Functional Description....92
14.5. Register Summary....94

15. Debug and Test Features....132
15.1. Description....132
15.2. Embedded Characteristics....132
15.3. Debug and Test Block Diagrams....133
15.4. Application Examples....134
15.5. Debug and Test Pin Description....136
15.6. Functional Description....136
15.7. Boundary JTAG ID Register....138
15.8. Cortex-A5 DP Identification Code Register IDCODE....138

16. Standard Boot Strategies....141
16.1. Description....141
16.2. Chip Access Using JTAG Connection....141
16.3. Flow Diagram....141
16.4. Chip Setup....142
16.5. Boot Configuration....142
16.6. SAM-BA Monitor....172
16.7. Fuse Box Controller....176

17. AXI Matrix (AXIMX)....178
17.1. Description....178
17.2. Embedded Characteristics....178
17.3. Operation....178
17.4. Register Summary....180

18. Matrix (H64MX/H32MX)....182
18.1. Description....182

18.2. Embedded Characteristics ..... 182
18.3. 64-bit Matrix (H64MX) ..... 182
18.4. 32-bit Matrix (H32MX) ..... 185
18.5. Memory Mapping ..... 187
18.6. Special Bus Granting Mechanism ..... 187
18.7. No Default Master ..... 188
18.8. Last Access Master ..... 188
18.9. Fixed Default Master ..... 188
18.10. Arbitration ..... 189
18.11. Register Write Protection ..... 191
18.12. TrustZone Technology ..... 192
18.13. Register Summary ..... 208

19. Special Function Registers (SFR) ..... 235
19.1. Description ..... 235
19.2. Embedded Characteristics ..... 235
19.3. Register Summary ..... 236

20. Special Function Registers Backup (SFRBU) ..... 254
20.1. Description ..... 254
20.2. Embedded Characteristics ..... 254
20.3. Register Summary ..... 255

21. Advanced Interrupt Controller (AIC) ..... 261
21.1. Description ..... 261
21.2. Embedded Characteristics ..... 261
21.3. Block Diagram ..... 262
21.4. Application Block Diagram ..... 262
21.5. AIC Detailed Block Diagram ..... 263
21.6. I/O Line Description ..... 263
21.7. Product Dependencies ..... 263
21.8. Functional Description ..... 264
21.9. Register Summary ..... 273

22. Watchdog Timer (WDT) ..... 297
22.1. Description ..... 297
22.2. Embedded Characteristics ..... 297
22.3. Block Diagram ..... 297
22.4. Functional Description ..... 298
22.5. Register Summary ..... 300

23. Reset Controller (RSTC) ..... 305
23.1. Description ..... 305
23.2. Embedded Characteristics ..... 305
23.3. Block Diagram ..... 305
23.4. Functional Description ..... 306
23.5. Register Summary ..... 311

24. Shutdown Controller (SHDWC)....316
24.1. Description....316
24.2. Embedded Characteristics....316
24.3. Block Diagram....316
24.4. I/O Lines Description....317
24.5. Product Dependencies....317
24.6. Functional Description....317
24.7. Register Summary....319

25. Periodic Interval Timer (PIT)....326
25.1. Description....326
25.2. Embedded Characteristics....326
25.3. Block Diagram....326
25.4. Functional Description....326
25.5. Register Summary....328

26. Real-time Clock (RTC)....333
26.1. Description....333
26.2. Embedded Characteristics....333
26.3. Block Diagram....334
26.4. Product Dependencies....334
26.5. Functional Description....334
26.6. Register Summary....346

27. System Controller Write Protection (SYSCWP)....379
27.1. Functional Description....379
27.2. Register Summary....380

28. Slow Clock Controller (SCKC)....382
28.1. Description....382
28.2. Embedded Characteristics....382
28.3. Block Diagram....382
28.4. Functional Description....382
28.5. Register Summary....384

29. Peripheral Touch Controller (PTC)....386
29.1. Description....386
29.2. Embedded Characteristics....386
29.3. Block Diagram....387
29.4. Signal Description....387
29.5. Product Dependencies....388
29.6. Functional Description....390
29.7. Register Summary....395

30. Low Power Asynchronous Receiver (RXLP)....399
30.1. Description....399
30.2. Embedded Characteristics....399

30.3. Block Diagram....399
30.4. Product Dependencies....400
30.5. Functional Description....400
30.6. Register Summary....403

31. Clock Generator....410
31.1. Description....410
31.2. Embedded Characteristics....410
31.3. Block Diagram....411
31.4. Slow Clock....411
31.5. Main Clock....412
31.6. Divider and PLLA Block....416
31.7. UTMI PLL Clock....417
31.8. Audio PLL....418

32. Power Management Controller (PMC)....420
32.1. Description....420
32.2. Embedded Characteristics....420
32.3. Block Diagram....421
32.4. Master Clock Controller....422
32.5. Processor Clock Controller....422
32.6. Matrix Clock Controller....423
32.7. Programmable Clock Controller....423
32.8. Core and Bus Independent Clocks for Peripherals....423
32.9. Peripheral and Generic Clock Controller....424
32.10. LCDC Clock Controller....425
32.11. ISC Clock Controller....425
32.12. USB Device and Host Clocks....425
32.13. DDR2/LPDDR/LPDDR2 Clock Controller....426
32.14. Fast Start-up from Ultra-Low-Power (ULP) Mode 0....426
32.15. Fast Start-up from Ultra-Low-Power (ULP) Mode 1....427
32.16. Asynchronous Partial Wake-up (SleepWalking)....429
32.17. Main Crystal Oscillator Failure Detection....431
32.18. 32.768 kHz Crystal Oscillator Frequency Monitor....432
32.19. Programming Sequence....433
32.20. Clock Switching Details....435
32.21. Register Write Protection....438
32.22. Register Summary....440

33. Parallel Input/Output Controller (PIO)....498
33.1. Description....498
33.2. Embedded Characteristics....498
33.3. Block Diagram....499
33.4. Product Dependencies....500
33.5. Functional Description....500
33.6. I/O Lines Programming Example....512
33.7. Register Summary....515

34. External Memories....565
34.1. Multiport DDR-SDRAM Controller (MPDDRC)....565
34.2. External Bus Interface (EBI)....573

35. AHB Multiport DDR-SDRAM Controller (MPDDRC)....576
35.1. Description....576
35.2. Embedded Characteristics....576
35.3. Block Diagram....577
35.4. Product Dependencies, Initialization Sequence....578
35.5. Functional Description....587
35.6. Software Interface/SDRAM Organization, Address Mapping....602
35.7. Register Summary....610

36. Static Memory Controller (SMC)....663
36.1. Description....663
36.2. Embedded Characteristics....663
36.3. Block Diagram....664
36.4. I/O Lines Description....664
36.5. Multiplexed Signals....665
36.6. Application Example....666
36.7. Product Dependencies....666
36.8. External Memory Mapping....666
36.9. Connection to External Devices....667
36.10. Standard Read and Write Protocols....669
36.11. Scrambling/Unscrambling Function....675
36.12. Automatic Wait States....676
36.13. Data Float Wait States....679
36.14. External Wait....683
36.15. Slow Clock Mode....687
36.16. Register Write Protection....689
36.17. NFC Operations....690
36.18. PMECC Controller Functional Description....700
36.19. Software Implementation....707
36.20. Register Summary....712

37. DMA Controller (XDMAC)....768
37.1. Description....768
37.2. Embedded Characteristics....768
37.3. Block Diagram....769
37.4. DMA Controller Peripheral Connections....769
37.5. Functional Description....773
37.6. Linked List Descriptor Operation....779
37.7. XDMAC Maintenance Software Operations....781
37.8. XDMAC Software Requirements....782
37.9. Register Summary....783

38. LCD Controller (LCDC)....845

38.1. Description ... 845
38.2. Embedded Characteristics ... 845
38.3. Block Diagram ... 846
38.4. I/O Lines Description ... 846
38.5. Product Dependencies ... 847
38.6. Functional Description ... 847
38.7. Register Summary ... 880

39. Ethernet MAC (GMAC) ... 1090
39.1. Description ... 1090
39.2. Embedded Characteristics ... 1090
39.3. Block Diagram ... 1091
39.4. Signal Interfaces ... 1091
39.5. Product Dependencies ... 1092
39.6. Functional Description ... 1092
39.7. Programming Interface ... 1123
39.8. Register Summary ... 1128

40. USB High Speed Device Port (UDPHS) ... 1285
40.1. Description ... 1285
40.2. Embedded Characteristics ... 1285
40.3. Block Diagram ... 1286
40.4. Typical Connection ... 1287
40.5. Product Dependencies ... 1287
40.6. Functional Description ... 1287
40.7. Register Summary ... 1312

41. USB Host High Speed Port (UHPHS) ... 1400
41.1. Description ... 1400
41.2. Embedded Characteristics ... 1400
41.3. Block Diagram ... 1401
41.4. Typical Connection ... 1402
41.5. Product Dependencies ... 1402
41.6. Functional Description ... 1404
41.7. Register Summary ... 1406

42. Audio Class D Amplifier (CLASSD) ... 1435
42.1. Description ... 1435
42.2. Embedded Characteristics ... 1435
42.3. Block Diagram ... 1436
42.4. Pin Name List ... 1436
42.5. Product Dependencies ... 1437
42.6. Functional Description ... 1437
42.7. Register Summary ... 1451

43. Inter-IC Sound Controller (I2SC) ... 1466
43.1. Description ... 1466
43.2. Embedded Characteristics ... 1466

43.3. Block Diagram....1467
43.4. I/O Lines Description....1467
43.5. Product Dependencies....1467
43.6. Functional Description....1468
43.7. I2SC Application Examples....1473
43.8. Register Summary....1476

44. Synchronous Serial Controller (SSC)....1492
44.1. Description....1492
44.2. Embedded Characteristics....1492
44.3. Block Diagram....1493
44.4. Application Block Diagram....1493
44.5. SSC Application Examples....1493
44.6. Pin Name List....1495
44.7. Product Dependencies....1495
44.8. Functional Description....1496
44.9. Register Summary....1507

45. Two-wire Interface (TWIHS)....1536
45.1. Description....1536
45.2. Embedded Characteristics....1536
45.3. List of Abbreviations....1537
45.4. Block Diagram....1537
45.5. Product Dependencies....1538
45.6. Functional Description....1538
45.7. Register Summary....1587

46. Flexible Serial Communication Controller (FLEXCOM)....1641
46.1. Description....1641
46.2. Embedded Characteristics....1642
46.3. Block Diagram....1645
46.4. I/O Lines Description....1645
46.5. Product Dependencies....1645
46.6. Register Accesses....1646
46.7. USART Functional Description....1646
46.8. SPI Functional Description....1696
46.9. TWI Functional Description....1717
46.10. Register Summary....1764

47. Universal Asynchronous Receiver Transmitter (UART)....1924
47.1. Description....1924
47.2. Embedded Characteristics....1924
47.3. Block Diagram....1924
47.4. Product Dependencies....1925
47.5. Functional Description....1925
47.6. Register Summary....1936

48. Serial Peripheral Interface (SPI)....1954

48.1. Description.... 1954
48.2. Embedded Characteristics.... 1954
48.3. Block Diagram.... 1955
48.4. Application Block Diagram.... 1955
48.5. Signal Description.... 1956
48.6. Product Dependencies.... 1956
48.7. Functional Description.... 1956
48.8. Register Summary.... 1978

49. Quad Serial Peripheral Interface (QSPI).... 2010
49.1. Description.... 2010
49.2. Embedded Characteristics.... 2010
49.3. Block Diagram.... 2011
49.4. Signal Description.... 2011
49.5. Product Dependencies.... 2011
49.6. Functional Description.... 2012
49.7. Register Summary.... 2028

50. Secure Digital MultiMedia Card Controller (SDMMC).... 2052
50.1. Description.... 2052
50.2. Embedded Characteristics.... 2052
50.3. Reference Documents.... 2053
50.4. Block Diagram.... 2054
50.5. Application Block Diagram.... 2055
50.6. Pin Name List.... 2055
50.7. Product Dependencies.... 2056
50.8. SD/SDIO Operating Mode.... 2056
50.9. e.MMC Operating Mode.... 2056
50.10. SDR104 / HS200 Tuning.... 2058
50.11. I/O Calibration.... 2061
50.12. Register Summary.... 2062

51. Image Sensor Controller (ISC).... 2171
51.1. Description.... 2171
51.2. Embedded Characteristics.... 2171
51.3. Block Diagram and Use Cases.... 2172
51.4. Product Dependencies.... 2174
51.5. Functional Description.... 2174
51.6. Register Summary.... 2196

52. Controller Area Network (MCAN).... 2275
52.1. Description.... 2275
52.2. Embedded Characteristics.... 2275
52.3. Block Diagram.... 2276
52.4. Product Dependencies.... 2276
52.5. Functional Description.... 2277
52.6. Register Summary.... 2305

53. Timer Counter (TC)....2379
53.1. Description....2379
53.2. Embedded Characteristics....2379
53.3. Block Diagram....2380
53.4. Pin List....2381
53.5. Product Dependencies....2381
53.6. Functional Description....2382
53.7. Register Summary....2404
54. Pulse Density Modulation Interface Controller (PDMIC)....2444
54.1. Description....2444
54.2. Embedded Characteristics....2444
54.3. Block Diagram....2444
54.4. Signal Description....2444
54.5. Product Dependencies....2445
54.6. Functional Description....2445
54.7. Register Summary....2451
55. Pulse Width Modulation Controller (PWM)....2465
55.1. Description....2465
55.2. Embedded Characteristics....2465
55.3. Block Diagram....2467
55.4. I/O Lines Description....2467
55.5. Product Dependencies....2468
55.6. Functional Description....2469
55.7. Register Summary....2511
56. Secure Fuse Controller (SFC)....2585
56.1. Description....2585
56.2. Embedded Characteristics....2585
56.3. Block Diagram....2586
56.4. Functional Description....2586
56.5. Register Summary....2588
57. Integrity Check Monitor (ICM)....2597
57.1. Description....2597
57.2. Embedded Characteristics....2598
57.3. Block Diagram....2599
57.4. Product Dependencies....2599
57.5. Functional Description....2600
57.6. Register Summary....2614
58. Advanced Encryption Standard Bridge (AESB)....2635
58.1. Description....2635
58.2. Embedded Characteristics....2635
58.3. Product Dependencies....2635
58.4. Functional Description....2636

58.5. Register Summary....2640

59. Advanced Encryption Standard (AES).... 2655
59.1. Description.... 2655
59.2. Embedded Characteristics.... 2655
59.3. Product Dependencies....2656
59.4. Functional Description....2656
59.5. Register Summary....2678

60. Secure Hash Algorithm (SHA)....2707
60.1. Description.... 2707
60.2. Embedded Characteristics.... 2707
60.3. Product Dependencies....2707
60.4. Functional Description....2707
60.5. Register Summary....2716

61. Triple Data Encryption Standard (TDES)....2732
61.1. Description.... 2732
61.2. Embedded Characteristics.... 2732
61.3. Product Dependencies....2733
61.4. Functional Description....2733
61.5. Register Summary....2739

62. True Random Number Generator (TRNG).... 2757
62.1. Description.... 2757
62.2. Embedded Characteristics.... 2757
62.3. Block Diagram....2757
62.4. Product Dependencies....2757
62.5. Functional Description....2758
62.6. Register Summary....2759

63. Analog Comparator Controller (ACC)....2766
63.1. Description.... 2766
63.2. Embedded Characteristics.... 2766
63.3. Block Diagram....2766
63.4. Signal Description.... 2766
63.5. Product Dependencies....2766
63.6. Functional Description....2767
63.7. Register Summary....2768

64. Security Module (SECUMOD)....2773
64.1. Description.... 2773
64.2. Embedded Characteristics.... 2773
64.3. Block Diagram....2774
64.4. Product Dependencies....2775
64.5. Functional Description....2775
64.6. Register Summary....2782

65. Analog-to-Digital Controller (ADC)..... 2806
65.1. Description..... 2806
65.2. Embedded Characteristics..... 2806
65.3. Block Diagram..... 2807
65.4. Signal Description..... 2808
65.5. Product Dependencies..... 2808
65.6. Functional Description..... 2808
65.7. Register Summary..... 2836

66. Electrical Characteristics..... 2881
66.1. Absolute Maximum Ratings..... 2881
66.2. DC Characteristics..... 2881
66.3. Power Consumption..... 2884
66.4. Active Mode..... 2885
66.5. Low-power Modes..... 2887
66.6. Clock Characteristics..... 2892
66.7. Oscillator Characteristics..... 2893
66.8. PLL Characteristics..... 2897
66.9. USB HS Characteristics..... 2898
66.10. PTC Characteristics..... 2898
66.11. ADC Characteristics..... 2898
66.12. Analog Comparator Characteristics..... 2905
66.13. POR Characteristics..... 2905
66.14. SMC Timings..... 2906
66.15. FLEXCOM Timings..... 2911
66.16. USART in Asynchronous Modes..... 2919
66.17. SPI Timings..... 2919
66.18. TWI Timings..... 2926
66.19. QSPI Timings..... 2927
66.20. MPDDRC Timings..... 2932
66.21. SSC Timings..... 2933
66.22. PDMIC Timings..... 2940
66.23. I2SC Timings..... 2941
66.24. ISC Timings..... 2943
66.25. SDMMC Timings..... 2944
66.26. GMAC Timings..... 2945

67. Mechanical Characteristics..... 2948
67.1. 289-Ball Low Profile Fine Pitch Ball Grid Array (AMB) - 14x14x1.4 mm Body [LFBGA] Atmel Legacy Global Package Code CCZ..... 2948
67.2. 256-Ball Thin Fine Pitch Ball Grid Array (AYB) - 8x8x1.05 mm Body [TFBGA]..... 2952
67.3. 196-Ball Thin Fine Pitch Ball Grid Array (BAB) - 11x11 mm Body [TFBGA]..... 2956

68. Schematic Checklist..... 2960
68.1. Power Supply..... 2960
68.2. Power-On Reset..... 2966
68.3. Clock, Oscillator and PLL..... 2966

© 2018 Microchip Technology Inc.

68.4. ICE and JTAG.... 2968
68.5. Reset and Test.... 2970
68.6. Shutdown/Wake-up Logic.... 2971
68.7. Parallel Input/Output (PIO).... 2971
68.8. Analog-to-Digital Converter (ADC).... 2971
68.9. External Bus Interface (EBI).... 2972
68.10. USB High-Speed Host Port (UHPHS) / USB High-Speed Device Port (UDPHS).... 2973
68.11. Boot Program Hardware Constraints.... 2974
68.12. Layout and Design Constraints.... 2974

69. Marking.... 2978

70. Ordering Information.... 2979

71. Revision History.... 2980
71.1. Revision DS60001476C.... 2980
71.2. Revision DS60001476B.... 2984
71.3. Revision DS60001476A.... 2986
71.4. Revision 11267E.... 2994
71.5. Revision 11267D.... 2994
71.6. Revision 11267C.... 3003
71.7. Revision 11267B.... 3007
71.8. Revision 11267A.... 3010

The Microchip Web Site.... 3024

Customer Change Notification Service.... 3024

Customer Support.... 3024

Product Identification System.... 3025

Microchip Devices Code Protection Feature.... 3025

Legal Notice.... 3026

Trademarks.... 3026

Quality Management System Certified by DNV.... 3027

Worldwide Sales and Service.... 3028

## The Microchip Web Site

Microchip provides online support via our web site at http://www.microchip.com/. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

## Customer Change Notification Service

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at http://www.microchip.com/. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

## Customer Support

Users of Microchip products can receive assistance through several channels:

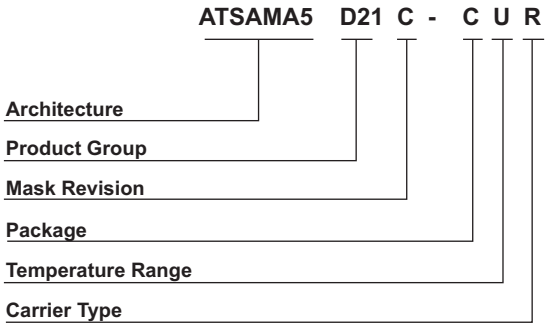
- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: http://www.microchip.com/support

## Product Identification System

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.




| | | |
|---|---|---|
| Architecture: | ATSAMA5 | = Arm Cortex-A5 CPU |
| Product Group: | D21, D22, D23 | = 196-ball general-purpose microprocessors |
| | D24 | = 256-ball general-purpose microprocessors |
| | D28 | = 289-ball general-purpose microprocessors |
| Mask Revision: | C | |
| Package: | C | = BGA |
| Temperature Range: | U | = -40°C to +85°C (Industrial) |
| | N | = -40°C to +105°C (Extended Industrial) |
| Carrier Type: | Blank | = Standard packaging (tray) |
| | R | = Tape and Reel |

Examples:

- ATSAMA5D21C-CUR = ARM Cortex-A5 general- purpose microprocessor, 196-ball, Industrial temperature, BGA Package.

**Note:** Tape and Reel identifier only appears in the catalog part number description. This identifier is used for ordering purposes and is not printed on the device package

## Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the

operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.

- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

## Legal Notice

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

## Trademarks

The Microchip name and logo, the Microchip logo, AnyRate, AVR, AVR logo, AVR Freaks, BitCloud, chipKIT, chipKIT logo, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, Heldo, JukeBlox, KeeLoq, Kleer, LANCheck, LINK MD, maXStylus, maXTouch, MediaLB, megaAVR, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, Prochip Designer, QTouch, SAM-BA, SpyNIC, SST, SST Logo, SuperFlash, tinyAVR, UNI/O, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ClockWorks, The Embedded Control Solutions Company, EtherSynch, Hyper Speed Control, HyperLight Load, IntelliMOS, mTouch, Precision Edge, and Quiet-Wire are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BodyCom, CodeGuard, CryptoAuthentication, CryptoAutomotive, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming, ICSP, INICnet, Inter-Chip Connectivity, JitterBlocker, KleerNet, KleerNet logo, memBrain, Mindi, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PowerSmart, PureSilicon, QMatrix, REAL ICE, Ripple Blocker, SAM-ICE, Serial Quad I/O, SMART-I.S., SQI, SuperSwitcher, SuperSwitcher II, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.

© 2018, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

ISBN: 978-1-5224-3608-9

AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, µVision, Versatile are trademarks or registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

## Quality Management System Certified by DNV

**ISO/TS 16949**

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

# Worldwide Sales and Service

| AMERICAS | ASIA/PACIFIC | ASIA/PACIFIC | EUROPE |
|---|---|---|---|
| **Corporate Office**<br>2355 West Chandler Blvd.<br>Chandler, AZ 85224-6199<br>Tel: 480-792-7200<br>Fax: 480-792-7277<br>Technical Support:<br>http://www.microchip.com/support<br>Web Address:<br>www.microchip.com<br>**Atlanta**<br>Duluth, GA<br>Tel: 678-957-9614<br>Fax: 678-957-1455<br>**Austin, TX**<br>Tel: 512-257-3370<br>**Boston**<br>Westborough, MA<br>Tel: 774-760-0087<br>Fax: 774-760-0088<br>**Chicago**<br>Itasca, IL<br>Tel: 630-285-0071<br>Fax: 630-285-0075<br>**Dallas**<br>Addison, TX<br>Tel: 972-818-7423<br>Fax: 972-818-2924<br>**Detroit**<br>Novi, MI<br>Tel: 248-848-4000<br>**Houston, TX**<br>Tel: 281-894-5983<br>**Indianapolis**<br>Noblesville, IN<br>Tel: 317-773-8323<br>Fax: 317-773-5453<br>Tel: 317-536-2380<br>**Los Angeles**<br>Mission Viejo, CA<br>Tel: 949-462-9523<br>Fax: 949-462-9608<br>Tel: 951-273-7800<br>**Raleigh, NC**<br>Tel: 919-844-7510<br>**New York, NY**<br>Tel: 631-435-6000<br>**San Jose, CA**<br>Tel: 408-735-9110<br>Tel: 408-436-4270<br>**Canada - Toronto**<br>Tel: 905-695-1980<br>Fax: 905-695-2078 | **Australia - Sydney**<br>Tel: 61-2-9868-6733<br>**China - Beijing**<br>Tel: 86-10-8569-7000<br>**China - Chengdu**<br>Tel: 86-28-8665-5511<br>**China - Chongqing**<br>Tel: 86-23-8980-9588<br>**China - Dongguan**<br>Tel: 86-769-8702-9880<br>**China - Guangzhou**<br>Tel: 86-20-8755-8029<br>**China - Hangzhou**<br>Tel: 86-571-8792-8115<br>**China - Hong Kong SAR**<br>Tel: 852-2943-5100<br>**China - Nanjing**<br>Tel: 86-25-8473-2460<br>**China - Qingdao**<br>Tel: 86-532-8502-7355<br>**China - Shanghai**<br>Tel: 86-21-3326-8000<br>**China - Shenyang**<br>Tel: 86-24-2334-2829<br>**China - Shenzhen**<br>Tel: 86-755-8864-2200<br>**China - Suzhou**<br>Tel: 86-186-6233-1526<br>**China - Wuhan**<br>Tel: 86-27-5980-5300<br>**China - Xian**<br>Tel: 86-29-8833-7252<br>**China - Xiamen**<br>Tel: 86-592-2388138<br>**China - Zhuhai**<br>Tel: 86-756-3210040 | **India - Bangalore**<br>Tel: 91-80-3090-4444<br>**India - New Delhi**<br>Tel: 91-11-4160-8631<br>**India - Pune**<br>Tel: 91-20-4121-0141<br>**Japan - Osaka**<br>Tel: 81-6-6152-7160<br>**Japan - Tokyo**<br>Tel: 81-3-6880- 3770<br>**Korea - Daegu**<br>Tel: 82-53-744-4301<br>**Korea - Seoul**<br>Tel: 82-2-554-7200<br>**Malaysia - Kuala Lumpur**<br>Tel: 60-3-7651-7906<br>**Malaysia - Penang**<br>Tel: 60-4-227-8870<br>**Philippines - Manila**<br>Tel: 63-2-634-9065<br>**Singapore**<br>Tel: 65-6334-8870<br>**Taiwan - Hsin Chu**<br>Tel: 886-3-577-8366<br>**Taiwan - Kaohsiung**<br>Tel: 886-7-213-7830<br>**Taiwan - Taipei**<br>Tel: 886-2-2508-8600<br>**Thailand - Bangkok**<br>Tel: 66-2-694-1351<br>**Vietnam - Ho Chi Minh**<br>Tel: 84-28-5448-2100 | **Austria - Wels**<br>Tel: 43-7242-2244-39<br>Fax: 43-7242-2244-393<br>**Denmark - Copenhagen**<br>Tel: 45-4450-2828<br>Fax: 45-4485-2829<br>**Finland - Espoo**<br>Tel: 358-9-4520-820<br>**France - Paris**<br>Tel: 33-1-69-53-63-20<br>Fax: 33-1-69-30-90-79<br>**Germany - Garching**<br>Tel: 49-8931-9700<br>**Germany - Haan**<br>Tel: 49-2129-3766400<br>**Germany - Heilbronn**<br>Tel: 49-7131-67-3636<br>**Germany - Karlsruhe**<br>Tel: 49-721-625370<br>**Germany - Munich**<br>Tel: 49-89-627-144-0<br>Fax: 49-89-627-144-44<br>**Germany - Rosenheim**<br>Tel: 49-8031-354-560<br>**Israel - Ra'anana**<br>Tel: 972-9-744-7705<br>**Italy - Milan**<br>Tel: 39-0331-742611<br>Fax: 39-0331-466781<br>**Italy - Padova**<br>Tel: 39-049-7625286<br>**Netherlands - Drunen**<br>Tel: 31-416-690399<br>Fax: 31-416-690340<br>**Norway - Trondheim**<br>Tel: 47-7289-7561<br>**Poland - Warsaw**<br>Tel: 48-22-3325737<br>**Romania - Bucharest**<br>Tel: 40-21-407-87-50<br>**Spain - Madrid**<br>Tel: 34-91-708-08-90<br>Fax: 34-91-708-08-91<br>**Sweden - Gothenberg**<br>Tel: 46-31-704-60-40<br>**Sweden - Stockholm**<br>Tel: 46-8-5090-4654<br>**UK - Wokingham**<br>Tel: 44-118-921-5800<br>Fax: 44-118-921-5820 |

# Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Microchip:

ATSAMA5D21A-CU ATSAMA5D21A-CUR ATSAMA5D22C-CNR ATSAMA5D22C-CUR ATSAMA5D27C-CUR ATSAMA5D26C-CN ATSAMA5D24C-CUR ATSAMA5D26C-CNR ATSAMA5D21C-CU ATSAMA5D22C-CN ATSAMA5D22C-CU ATSAMA5D24C-CU ATSAMA5D26C-CUR ATSAMA5D21C-CUR ATSAMA5D27C-CN ATSAMA5D27C-CU ATSAMA5D27C-CNR ATSAMA5D26C-CU ATSAMA5D21B-CU ATSAMA5D21BCUR

Общество с ограниченной ответственностью «МосЧип» ИНН 7719860671 / КПП 771901001
Адрес: 105318, г.Москва, ул.Щербаковская д.3, офис 1107

**Данный компонент на территории Российской Федерации**

**Вы можете приобрести в компании MosChip.**

Для оперативного оформления запроса Вам необходимо перейти по данной ссылке:

http://moschip.ru/get-element

Вы можете разместить у нас заказ для любого Вашего проекта, будь то серийное производство или разработка единичного прибора.

В нашем ассортименте представлены ведущие мировые производители активных и пассивных электронных компонентов.

Нашей специализацией является поставка электронной компонентной базы двойного назначения, продукции таких производителей как XILINX, Intel (ex.ALTERA), Vicor, Microchip, Texas Instruments, Analog Devices, Mini-Circuits, Amphenol, Glenair.

Сотрудничество с глобальными дистрибьюторами электронных компонентов, предоставляет возможность заказывать и получать с международных складов практически любой перечень компонентов в оптимальные для Вас сроки.

На всех этапах разработки и производства наши партнеры могут получить квалифицированную поддержку опытных инженеров.

Система менеджмента качества компании отвечает требованиям в соответствии с ГОСТ Р ИСО 9001, ГОСТ РВ 0015-002 и ЭС РД 009

**Офис по работе с юридическими лицами:**

105318, г.Москва, ул.Щербаковская д.3, офис 1107, 1118, ДЦ «Щербаковский»

Телефон: +7 495 668-12-70 (многоканальный)

Факс: +7 495 668-12-70 (доб.304)

E-mail: info@moschip.ru

Skype отдела продаж:
moschip.ru
moschip.ru_4
moschip.ru_6
moschip.ru_9