# nRF52811

**Product Specification**

v1.0



NORDIC® SEMICONDUCTOR

# Feature list

**Features:**

- *Bluetooth*® 5.1, IEEE 802.15.4-2006, 2.4 GHz transceiver
  - -97 dBm sensitivity in 1 Mbps *Bluetooth*® low energy mode
  - -104 dBm sensitivity in 125 kbps *Bluetooth*® low energy mode (long range)
  - -20 to +4 dBm TX power, configurable in 4 dB steps
  - On-air compatible with nRF52, nRF51, nRF24L, and nRF24AP Series
  - Supported data rates:
    - *Bluetooth*® 5.1: 2 Mbps, 1 Mbps, 500 kbps, and 125 kbps
    - IEEE 802.15.4-2006: 250 kbps
    - Proprietary 2.4 GHz: 2 Mbps, 1 Mbps
  - Angle-of-arrival (AoA) and angle-of-departure (AoD) direction finding using *Bluetooth*®.
  - Single-ended antenna output (on-chip balun)
  - 4.6 mA peak current in TX (0 dBm)
  - 4.6 mA peak current in RX
  - RSSI (1 dB resolution)
- ARM® Cortex®-M4 32-bit processor, 64 MHz
  - 144 EEMBC CoreMark® score running from flash memory
  - 34.4 µA/MHz running CoreMark from flash memory
  - 32.8 µA/MHz running CoreMark from RAM memory
  - Serial wire debug (SWD)
- Flexible power management
  - 1.7 V to 3.6 V supply voltage range
  - Fully automatic LDO and DC/DC regulator system
  - Fast wake-up using 64 MHz internal oscillator
  - 0.3 µA at 3 V in System OFF mode, no RAM retention
  - 0.5 µA at 3 V in System OFF mode with full 24 kB RAM retention
  - 1.5 µA at 3 V in System ON mode, with full 24 kB RAM retention, wake on RTC
  - 1.4 µA at 3 V in System ON mode, no RAM retention, wake on RTC
- 192 kB flash and 24 kB RAM
- Nordic SoftDevice ready
- Support for concurrent multi-protocol
- 12-bit, 200 ksps ADC - 8 configurable channels with programmable gain
- 64 level comparator
- Temperature sensor
- Up to 32 general purpose I/O pins
- 4-channel pulse width modulator (PWM) unit with EasyDMA
- Digital microphone interface (PDM)
- 3x 32-bit timer with counter mode
- 2x SPI master/slave with EasyDMA
- I2C compatible 2-wire master/slave
- UART (CTS/RTS) with EasyDMA
- Programmable peripheral interconnect (PPI)
- Quadrature decoder (QDEC)
- AES HW encryption with EasyDMA
- 2x real-time counter (RTC)
- Single crystal operation
- Package variants
  - QFN48 package, 6 x 6 mm
  - QFN32 package, 5 x 5 mm
  - WLCSP package, 2.482 x 2.464 mm

**Applications:**

- Computer peripherals and I/O devices
  - Mouse
  - Keyboard
  - Mobile HID
- CE remote controls
- Network processor
  - Wearables
  - Virtual reality headsets
- Health and medical
- Enterprise lighting
  - Industrial
  - Commercial
  - Retail
- Beacons
- Connectivity device in multi-chip solutions

# Contents


Feature list . . . ii

1 Revision history . . . 10

2 About this document . . . 11

2.1 Document naming and status . . . 11
2.2 Peripheral naming and abbreviations . . . 11
2.3 Register tables . . . 12
2.3.1 Fields and values . . . 12
2.4 Registers . . . 12
2.4.1 DUMMY . . . 12

3 Block diagram . . . 14

4 Core components . . . 15

4.1 CPU . . . 15
4.1.1 Electrical specification . . . 15
4.1.2 CPU and support module configuration . . . 15
4.2 Memory . . . 16
4.2.1 RAM - Random access memory . . . 17
4.2.2 Flash - Non-volatile memory . . . 17
4.2.3 Memory map . . . 17
4.2.4 Instantiation . . . 18
4.3 NVMC — Non-volatile memory controller . . . 19
4.3.1 Writing to flash . . . 19
4.3.2 Erasing a page in flash . . . 19
4.3.3 Writing to user information configuration registers (UICR) . . . 20
4.3.4 Erasing user information configuration registers (UICR) . . . 20
4.3.5 Erase all . . . 20
4.3.6 Partial erase of a page in flash . . . 20
4.3.7 Registers . . . 20
4.3.8 Electrical specification . . . 24
4.4 FICR — Factory information configuration registers . . . 24
4.4.1 Registers . . . 24
4.5 UICR — User information configuration registers . . . 32
4.5.1 Registers . . . 32
4.6 EasyDMA . . . 35
4.6.1 EasyDMA error handling . . . 36
4.6.2 EasyDMA array list . . . 37
4.7 AHB multilayer . . . 37
4.8 Debug . . . 38
4.8.1 DAP - Debug access port . . . 39
4.8.2 CTRL-AP - Control access port . . . 39
4.8.3 Debug interface mode . . . 41
4.8.4 Real-time debug . . . 41

5 Power and clock management . . . 42

5.1 Power management unit (PMU) . . . 42
5.2 Current consumption . . . 42

5.2.1 Electrical specification . . . 43
5.3 POWER — Power supply . . . 48
5.3.1 Regulators . . . 48
5.3.2 System OFF mode . . . 49
5.3.3 System ON mode . . . 50
5.3.4 Power supply supervisor . . . 50
5.3.5 RAM power control . . . 52
5.3.6 Reset . . . 52
5.3.7 Retained registers . . . 53
5.3.8 Reset behavior . . . 53
5.3.9 Registers . . . 54
5.3.10 Electrical specification . . . 60
5.4 CLOCK — Clock control . . . 61
5.4.1 HFCLK clock controller . . . 62
5.4.2 LFCLK clock controller . . . 63
5.4.3 Registers . . . 66
5.4.4 Electrical specification . . . 72

6 Peripherals . . . 74

6.1 Peripheral interface . . . 74
6.1.1 Peripheral ID . . . 74
6.1.2 Peripherals with shared ID . . . 75
6.1.3 Peripheral registers . . . 75
6.1.4 Bit set and clear . . . 75
6.1.5 Tasks . . . 75
6.1.6 Events . . . 75
6.1.7 Shortcuts . . . 76
6.1.8 Interrupts . . . 76
6.2 AAR — Accelerated address resolver . . . 77
6.2.1 EasyDMA . . . 77
6.2.2 Resolving a resolvable address . . . 77
6.2.3 Use case example for chaining RADIO packet reception with address resolution using AAR . . . 78
6.2.4 IRK data structure . . . 78
6.2.5 Registers . . . 78
6.2.6 Electrical specification . . . 82
6.3 BPROT — Block protection . . . 82
6.3.1 Registers . . . 84
6.4 CCM — AES CCM mode encryption . . . 85
6.4.1 Key-steam generation . . . 85
6.4.2 Encryption . . . 86
6.4.3 Decryption . . . 86
6.4.4 AES CCM and RADIO concurrent operation . . . 87
6.4.5 Encrypting packets on-the-fly in radio transmit mode . . . 87
6.4.6 Decrypting packets on-the-fly in radio receive mode . . . 88
6.4.7 CCM data structure . . . 89
6.4.8 EasyDMA and ERROR event . . . 90
6.4.9 Registers . . . 90
6.4.10 Electrical specification . . . 97
6.5 COMP — Comparator . . . 97
6.5.1 Differential mode . . . 99
6.5.2 Single-ended mode . . . 99
6.5.3 Registers . . . 101
6.5.4 Electrical specification . . . 108
6.6 ECB — AES electronic codebook mode encryption . . . 109

6.6.1 Shared resources . . . 109
6.6.2 EasyDMA . . . 109
6.6.3 ECB data structure . . . 109
6.6.4 Registers . . . 110
6.6.5 Electrical specification . . . 112
6.7 EGU — Event generator unit . . . 112
6.7.1 Registers . . . 113
6.7.2 Electrical specification . . . 115
6.8 GPIO — General purpose input/output . . . 115
6.8.1 Pin configuration . . . 116
6.8.2 Registers . . . 117
6.8.3 Electrical specification . . . 122
6.9 GPIOTE — GPIO tasks and events . . . 123
6.9.1 Pin events and tasks . . . 123
6.9.2 Port event . . . 124
6.9.3 Tasks and events pin configuration . . . 124
6.9.4 Registers . . . 125
6.9.5 Electrical specification . . . 129
6.10 PDM — Pulse density modulation interface . . . 129
6.10.1 Master clock generator . . . 129
6.10.2 Module operation . . . 129
6.10.3 Decimation filter . . . 130
6.10.4 EasyDMA . . . 130
6.10.5 Hardware example . . . 131
6.10.6 Pin configuration . . . 132
6.10.7 Registers . . . 132
6.10.8 Electrical specification . . . 138
6.11 PPI — Programmable peripheral interconnect . . . 139
6.11.1 Pre-programmed channels . . . 140
6.11.2 Registers . . . 141
6.12 PWM — Pulse width modulation . . . 145
6.12.1 Wave counter . . . 146
6.12.2 Decoder with EasyDMA . . . 149
6.12.3 Limitations . . . 156
6.12.4 Pin configuration . . . 156
6.12.5 Registers . . . 157
6.13 QDEC — Quadrature decoder . . . 165
6.13.1 Sampling and decoding . . . 166
6.13.2 LED output . . . 167
6.13.3 Debounce filters . . . 167
6.13.4 Accumulators . . . 168
6.13.5 Output/input pins . . . 168
6.13.6 Pin configuration . . . 168
6.13.7 Registers . . . 169
6.13.8 Electrical specification . . . 180
6.14 RADIO — 2.4 GHz radio . . . 180
6.14.1 Packet configuration . . . 181
6.14.2 Address configuration . . . 182
6.14.3 Data whitening . . . 182
6.14.4 CRC . . . 182
6.14.5 Radio states . . . 183
6.14.6 Transmit sequence . . . 184
6.14.7 Receive sequence . . . 185
6.14.8 Received signal strength indicator (RSSI) . . . 187

6.14.9 Interframe spacing (IFS) . . . 187
6.14.10 Device address match . . . 188
6.14.11 Bit counter . . . 188
6.14.12 Direction finding . . . 189
6.14.13 IEEE 802.15.4 operation . . . 194
6.14.14 EasyDMA . . . 202
6.14.15 Registers . . . 203
6.14.16 Electrical specification . . . 237
6.15 RNG — Random number generator . . . 243
6.15.1 Bias correction . . . 243
6.15.2 Speed . . . 243
6.15.3 Registers . . . 244
6.15.4 Electrical specification . . . 246
6.16 RTC — Real-time counter . . . 246
6.16.1 Clock source . . . 247
6.16.2 Resolution versus overflow and the PRESCALER . . . 247
6.16.3 COUNTER register . . . 248
6.16.4 Overflow features . . . 248
6.16.5 TICK event . . . 248
6.16.6 Event control feature . . . 249
6.16.7 Compare feature . . . 249
6.16.8 TASK and EVENT jitter/delay . . . 251
6.16.9 Reading the COUNTER register . . . 253
6.16.10 Registers . . . 254
6.16.11 Electrical specification . . . 259
6.17 SAADC — Successive approximation analog-to-digital converter . . . 259
6.17.1 Shared resources . . . 259
6.17.2 Overview . . . 259
6.17.3 Digital output . . . 260
6.17.4 Analog inputs and channels . . . 261
6.17.5 Operation modes . . . 261
6.17.6 EasyDMA . . . 263
6.17.7 Resistor ladder . . . 264
6.17.8 Reference . . . 265
6.17.9 Acquisition time . . . 265
6.17.10 Limits event monitoring . . . 266
6.17.11 Registers . . . 267
6.17.12 Electrical specification . . . 281
6.17.13 Performance factors . . . 282
6.18 SPI — Serial peripheral interface master . . . 282
6.18.1 Functional description . . . 283
6.18.2 Registers . . . 286
6.18.3 Electrical specification . . . 289
6.19 SPIM — Serial peripheral interface master with EasyDMA . . . 290
6.19.1 SPI master transaction sequence . . . 291
6.19.2 Master mode pin configuration . . . 292
6.19.3 EasyDMA . . . 293
6.19.4 Low power . . . 293
6.19.5 Registers . . . 293
6.19.6 Electrical specification . . . 302
6.20 SPIS — Serial peripheral interface slave with EasyDMA . . . 303
6.20.1 Shared resources . . . 304
6.20.2 EasyDMA . . . 304
6.20.3 SPI slave operation . . . 304

6.20.4 Pin configuration . . . 306
6.20.5 Registers . . . 307
6.20.6 Electrical specification . . . 318
6.21 SWI — Software interrupts . . . 320
6.21.1 Registers . . . 320
6.22 TEMP — Temperature sensor . . . 320
6.22.1 Registers . . . 320
6.22.2 Electrical specification . . . 326
6.23 TWI — I²C compatible two-wire interface . . . 327
6.23.1 Functional description . . . 327
6.23.2 Master mode pin configuration . . . 327
6.23.3 Shared resources . . . 328
6.23.4 Master write sequence . . . 328
6.23.5 Master read sequence . . . 329
6.23.6 Master repeated start sequence . . . 330
6.23.7 Low power . . . 331
6.23.8 Registers . . . 331
6.23.9 Electrical specification . . . 338
6.24 TIMER — Timer/counter . . . 339
6.24.1 Capture . . . 341
6.24.2 Compare . . . 341
6.24.3 Task delays . . . 341
6.24.4 Task priority . . . 341
6.24.5 Registers . . . 341
6.25 TWIM — I²C compatible two-wire interface master with EasyDMA . . . 346
6.25.1 EasyDMA . . . 347
6.25.2 Master write sequence . . . 348
6.25.3 Master read sequence . . . 349
6.25.4 Master repeated start sequence . . . 350
6.25.5 Low power . . . 351
6.25.6 Master mode pin configuration . . . 351
6.25.7 Registers . . . 351
6.25.8 Electrical specification . . . 362
6.25.9 Pullup resistor . . . 363
6.26 TWIS — I²C compatible two-wire interface slave with EasyDMA . . . 363
6.26.1 EasyDMA . . . 366
6.26.2 TWI slave responding to a read command . . . 366
6.26.3 TWI slave responding to a write command . . . 367
6.26.4 Master repeated start sequence . . . 368
6.26.5 Terminating an ongoing TWI transaction . . . 369
6.26.6 Low power . . . 369
6.26.7 Slave mode pin configuration . . . 369
6.26.8 Registers . . . 370
6.26.9 Electrical specification . . . 380
6.27 UART — Universal asynchronous receiver/transmitter . . . 380
6.27.1 Functional description . . . 381
6.27.2 Pin configuration . . . 381
6.27.3 Shared resources . . . 381
6.27.4 Transmission . . . 381
6.27.5 Reception . . . 382
6.27.6 Suspending the UART . . . 383
6.27.7 Error conditions . . . 383
6.27.8 Using the UART without flow control . . . 383
6.27.9 Parity and stop bit configuration . . . 383

6.27.10 Registers . . . 383
6.27.11 Electrical specification . . . 392
6.28 UARTE — Universal asynchronous receiver/transmitter with EasyDMA . . . 392
6.28.1 EasyDMA . . . 393
6.28.2 Transmission . . . 393
6.28.3 Reception . . . 394
6.28.4 Error conditions . . . 396
6.28.5 Using the UARTE without flow control . . . 396
6.28.6 Parity and stop bit configuration . . . 396
6.28.7 Low power . . . 396
6.28.8 Pin configuration . . . 397
6.28.9 Registers . . . 397
6.28.10 Electrical specification . . . 410
6.29 WDT — Watchdog timer . . . 410
6.29.1 Reload criteria . . . 411
6.29.2 Temporarily pausing the watchdog . . . 411
6.29.3 Watchdog reset . . . 411
6.29.4 Registers . . . 411
6.29.5 Electrical specification . . . 415

7 Hardware and layout . . . 416
7.1 Pin assignments . . . 416
7.1.1 QFN48 pin assignments . . . 416
7.1.2 QFN32 pin assignments . . . 418
7.1.3 WLCSP ball assignments . . . 420
7.1.4 GPIO pins located near the radio . . . 423
7.2 Mechanical specifications . . . 423
7.2.1 QFN48 6 x 6 mm package . . . 423
7.2.2 QFN32 5 x 5 mm package . . . 424
7.2.3 WLCSP 2.482 x 2.464 mm package . . . 424
7.3 Reference circuitry . . . 425
7.3.1 Schematic QFAA QFN48 with internal LDO regulator setup . . . 425
7.3.2 Schematic QFAA QFN48 with DC/DC regulator setup . . . 426
7.3.3 Schematic QCAA QFN32 with internal LDO regulator setup . . . 427
7.3.4 Schematic QCAA QFN32 with DC/DC regulator setup . . . 428
7.3.5 Schematic CAAA WLCSP with internal LDO regulator setup . . . 429
7.3.6 Schematic CAAA WLCSP with DC/DC regulator setup . . . 430
7.3.7 Schematic CAAA WLCSP with two layers . . . 432
7.3.8 PCB guidelines . . . 433
7.3.9 PCB layout example . . . 434

8 Recommended operating conditions . . . 436
8.1 WLCSP light sensitivity . . . 436

9 Absolute maximum ratings . . . 437

10 Ordering information . . . 438
10.1 IC marking . . . 438
10.2 Box labels . . . 438
10.3 Order code . . . 439
10.4 Code ranges and values . . . 440
10.5 Product options . . . 441

11 Legal notices . . . . . 443
11.1 Liability disclaimer . . . . . 443
11.2 Life support applications . . . . . 443
11.3 RoHS and REACH statement . . . . . 443
11.4 Trademarks . . . . . 443
11.5 Copyright notice . . . . . 444

# 1 Revision history

| Date | Version | Description |
|---|---|---|
| February 2019 | 1.0 | First release |

# 2 About this document

This product specification is organized into chapters based on the modules and peripherals that are available in this IC.

The peripheral descriptions are divided into separate sections that include the following information:

- A detailed functional description of the peripheral
- Register configuration for the peripheral
- Electrical specification tables, containing performance data which apply for the operating conditions described in Recommended operating conditions on page 436.

## 2.1 Document naming and status

Nordic uses three distinct names for this document, which are reflecting the maturity and the status of the document and its content.

| Document name | Description |
|---|---|
| Objective Product Specification (OPS) | Applies to document versions up to 0.7.<br>This product specification contains target specifications for product development. |
| Preliminary Product Specification (PPS) | Applies to document versions 0.7 and up to 1.0.<br>This product specification contains preliminary data. Supplementary data may be published from Nordic Semiconductor ASA later. |
| Product Specification (PS) | Applies to document versions 1.0 and higher.<br>This product specification contains final product specifications. Nordic Semiconductor ASA reserves the right to make changes at any time without notice in order to improve design and supply the best possible product. |

*Table 1: Defined document names*

## 2.2 Peripheral naming and abbreviations

Every peripheral has a unique capitalized name or an abbreviation of its name, e.g. TIMER, used for identification and reference. This name is used in chapter headings and references, and it will appear in the ARM® Cortex® Microcontroller Software Interface Standard (CMSIS) hardware abstraction layer to identify the peripheral.

The peripheral instance name, which is different from the peripheral name, is constructed using the peripheral name followed by a numbered postfix, starting with 0, for example, TIMER0. A postfix is normally only used if a peripheral can be instantiated more than once. The peripheral instance name is also used in the CMSIS to identify the peripheral instance.

# 2.3 Register tables

Individual registers are described using register tables. These tables are built up of two sections. The first three colored rows describe the position and size of the different fields in the register. The following rows describe the fields in more detail.

## 2.3.1 Fields and values

The **Id (Field Id)** row specifies the bits that belong to the different fields in the register. If a field has enumerated values, then every value will be identified with a unique value id in the **Value Id** column.

A blank space means that the field is reserved and read as undefined, and it also must be written as `0` to secure forward compatibility. If a register is divided into more than one field, a unique field name is specified for each field in the **Field** column. The **Value Id** may be omitted in the single-bit bit fields when values can be substituted with a Boolean type enumerator range, e.g. true/false, disable(d)/enable(d), on/off, and so on.

Values are usually provided as decimal or hexadecimal. Hexadecimal values have a `0x` prefix, decimal values have no prefix.

The **Value** column can be populated in the following ways:

- Individual enumerated values, for example 1, 3, 9.
- Range of values, e.g. [0..4], indicating all values from and including 0 and 4.
- Implicit values. If no values are indicated in the **Value** column, all bit combinations are supported, or alternatively the field's translation and limitations are described in the text instead.

If two or more fields are closely related, the **Value Id**, **Value**, and **Description** may be omitted for all but the first field. Subsequent fields will indicate inheritance with '..'.

A feature marked **Deprecated** should not be used for new designs.

# 2.4 Registers

| Register | Offset | Description |
|---|---|---|
| DUMMY | 0x514 | Example of a register controlling a dummy feature |

*Table 2: Register overview*

## 2.4.1 DUMMY

Address offset: 0x514

Example of a register controlling a dummy feature

| Bit number | | | | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | | | | | | | | D | D | D | D | | | | | C | C | C | | | | | | | | B | | | | | | | | A | A |
| **Reset 0x00050002** | | | | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** |
| ID | Acce | Field | Value ID | Value | Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| A | RW | FIELD_A | | | Example of a field with several enumerated values | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | Disabled | 0 | The example feature is disabled | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | NormalMode | 1 | The example feature is enabled in normal mode | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | ExtendedMode | 2 | The example feature is enabled along with extra functionality | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Bit number | | | | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | | | | | | | | D | D | D | D | | | | | C | C | C | | | | | | | | | B | | | | | | | A | A |
| **Reset 0x00050002** | | | | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** |
| ID | Acce | Field | Value ID | Value | | | | Description | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| B | RW | FIELD_B | | | | | | Example of a deprecated field | | | | | | | | | | | | | | | | | | | | | | | | | | | Deprecated |
| | | | Disabled | 0 | | | | The override feature is disabled | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | Enabled | 1 | | | | The override feature is enabled | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| C | RW | FIELD_C | | | | | | Example of a field with a valid range of values | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | ValidRange | [2..7] | | | | Example of allowed values for this field | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| D | RW | FIELD_D | | | | | | Example of a field with no restriction on the values | | | | | | | | | | | | | | | | | | | | | | | | | | | |

# 3 Block diagram

This block diagram illustrates the overall system. Arrows with white heads indicate signals that share physical pins with other signals.

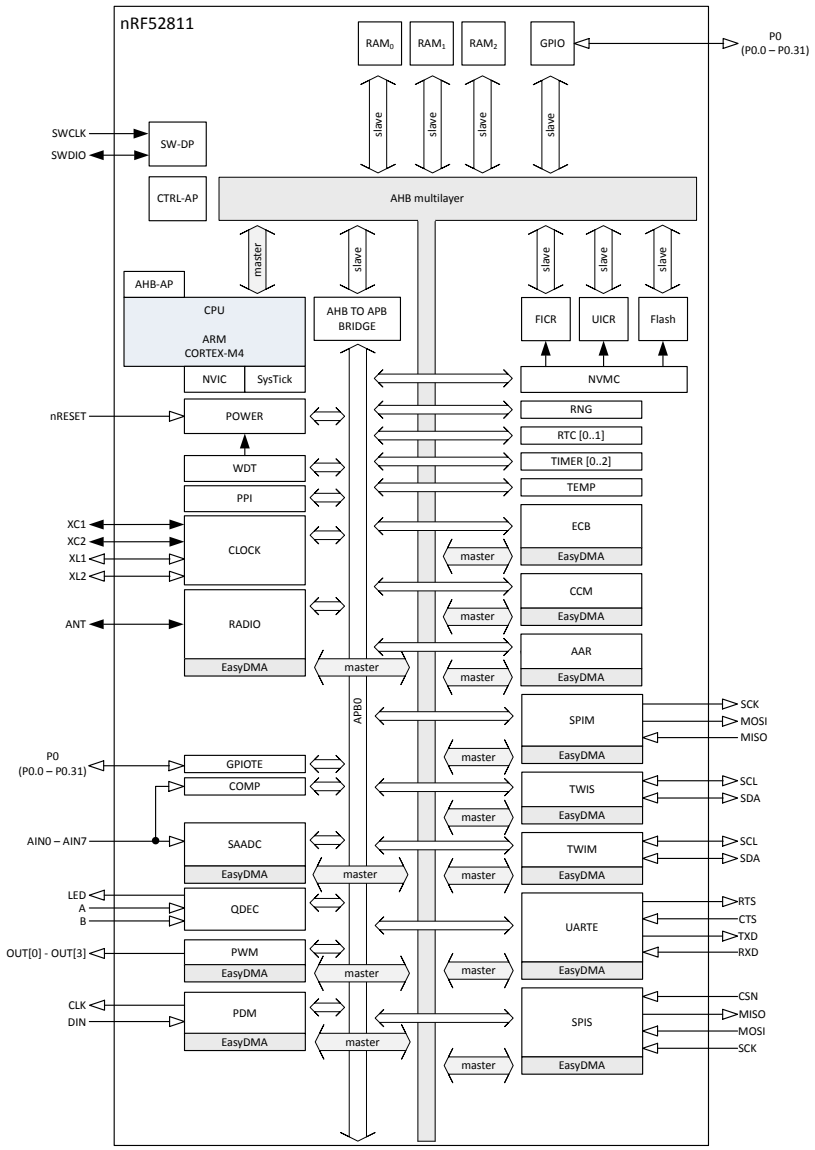



*Figure 1: Block diagram*

# 4 Core components

## 4.1 CPU

The ARM® Cortex®-M4 processor has a 32-bit instruction set (Thumb®-2 technology) that implements a superset of 16- and 32-bit instructions to maximize code density and performance.

This processor implements several features that enable energy-efficient arithmetic and high-performance signal processing including:

- Digital signal processing (DSP) instructions
- Single-cycle multiply and accumulate (MAC) instructions
- Hardware divide
- 8 and 16-bit single instruction multiple data (SIMD) instructions

The ARM Cortex Microcontroller Software Interface Standard (CMSIS) hardware abstraction layer for the ARM Cortex processor series is implemented and available for the M4 CPU.

Real-time execution is highly deterministic in thread mode, to and from sleep modes, and when handling events at configurable priority levels via the nested vectored interrupt controller (NVIC).

Executing code from flash will have a wait state penalty on the nRF52 Series. The section Electrical specification on page 15 shows CPU performance parameters including wait states in different modes, CPU current and efficiency, and processing power and efficiency based on the CoreMark® benchmark.

The ARM System Timer (SysTick) is present on the device. The SysTick's clock will only tick when the CPU is running or when the system is in debug interface mode.

### 4.1.1 Electrical specification

#### 4.1.1.1 CPU performance

The CPU clock speed is 64 MHz. Current and efficiency data is taken when in System ON and the CPU is executing the CoreMark® benchmark. It includes power regulator and clock base currents. All other blocks are IDLE.

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $W_{FLASH}$ | CPU wait states, running from flash | 0 | | 2 | |
| $W_{RAM}$ | CPU wait states, running from RAM | | | 0 | |
| $CM_{FLASH}$ | CoreMark[1], running from flash | | 144 | | CoreMark |
| $CM_{FLASH/MHz}$ | CoreMark per MHz, running from flash | | 2.25 | | CoreMark/MHz |
| $CM_{FLASH/mA}$ | CoreMark per mA, running from flash, DCDC 3V | | 65 | | CoreMark/mA |

### 4.1.2 CPU and support module configuration

The ARM® Cortex®-M4 processor has a number of CPU options and support modules implemented on the device.

[1] Using IAR v6.50.1.4452 with flags --endian=little --cpu=Cortex-M4 -e --fpu=VFPv4_sp –Ohs --no_size_constraints

| Option / Module | Description | Implemented |
|---|---|---|
| Core options | | |
| NVIC | Nested vector interrupt controller | 30 vectors |
| PRIORITIES | Priority bits | 3 |
| WIC | Wakeup interrupt controller | NO |
| Endianness | Memory system endianness | Little endian |
| Bit-banding | Bit banded memory | NO |
| DWT | Data watchpoint and trace | NO |
| SysTick | System tick timer | YES |
| Modules | | |
| MPU | Memory protection unit | YES |
| FPU | Floating-point unit | NO |
| DAP | Debug access port | YES |
| ETM | Embedded trace macrocell | NO |
| ITM | Instrumentation trace macrocell | NO |
| TPIU | Trace port interface unit | NO |
| ETB | Embedded trace buffer | NO |
| FPB | Flash patch and breakpoint unit | YES |
| HTM | AMBA® AHB trace macrocell | NO |

## 4.2 Memory

The nRF52811 contains flash and RAM that can be used for code and data storage.

The amount of RAM and flash will vary depending on variant, see Memory variants on page 16.

| Device name | RAM | Flash | Comments |
|---|---|---|---|
| nRF52811-QFAA | 24 kB | 192 kB | |
| nRF52811-QCAA | 24 kB | 192 kB | |
| nRF52811-CAAA | 24 kB | 192 kB | |

*Table 3: Memory variants*

The CPU and peripherals with EasyDMA can access memory via the AHB multilayer interconnect. The CPU is also able to access peripherals via the AHB multilayer interconnect, as illustrated in Memory layout on page 17.

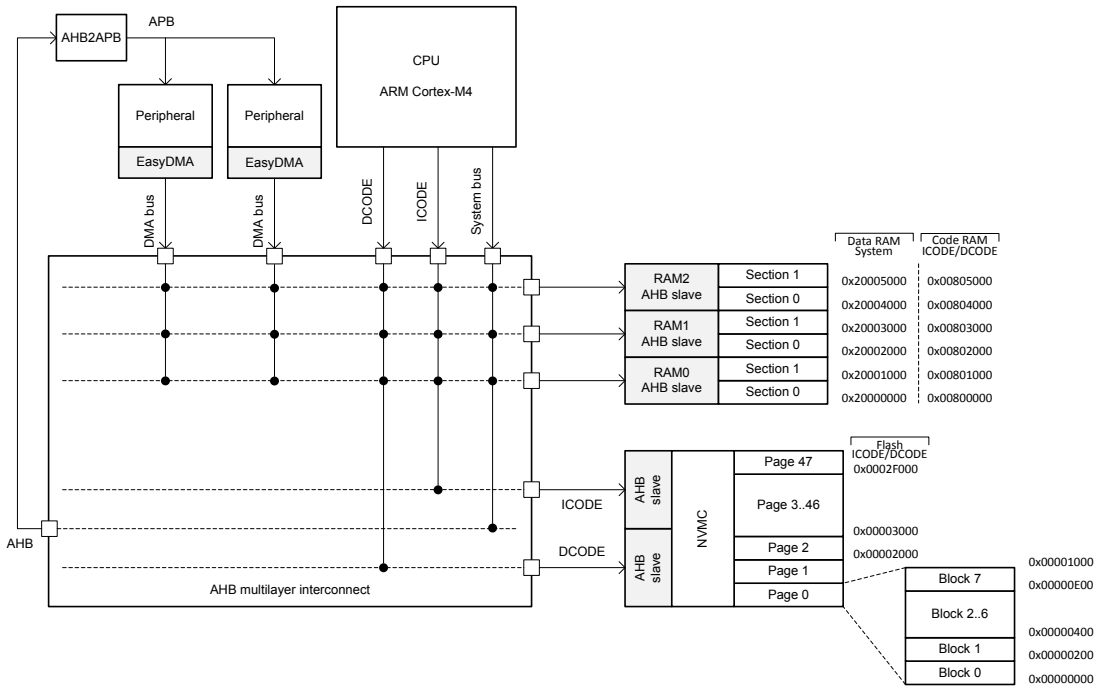



*Figure 2: Memory layout*

See AHB multilayer on page 37 and EasyDMA on page 35 for more information about the AHB multilayer interconnect and the EasyDMA.

The same physical RAM is mapped to both the Data RAM region and the Code RAM region. It is up to the application to partition the RAM within these regions so that one does not corrupt the other.

## 4.2.1 RAM - Random access memory

The RAM interface is divided into multiple RAM AHB slaves.

Each RAM AHB slave is connected to two 4-kilobyte RAM sections, see Section 0 and Section 1 in Memory layout on page 17.

Each of the RAM sections have separate power control for System ON and System OFF mode operation, which is configured via RAM register (see the POWER — Power supply on page 48).

## 4.2.2 Flash - Non-volatile memory

The flash can be read an unlimited number of times by the CPU, but it has restrictions on the number of times it can be written and erased, and also on how it can be written.

Writing to flash is managed by the non-volatile memory controller (NVMC), see NVMC — Non-volatile memory controller on page 19.

The flash is divided into multiple 4 kB pages that can be accessed by the CPU via both the ICODE and DCODE buses as shown in, Memory layout on page 17. Each page is divided into 8 blocks.

## 4.2.3 Memory map

The complete memory map is shown in Memory map on page 18. As described in Memory on page 16, Code RAM and Data RAM are the same physical RAM.

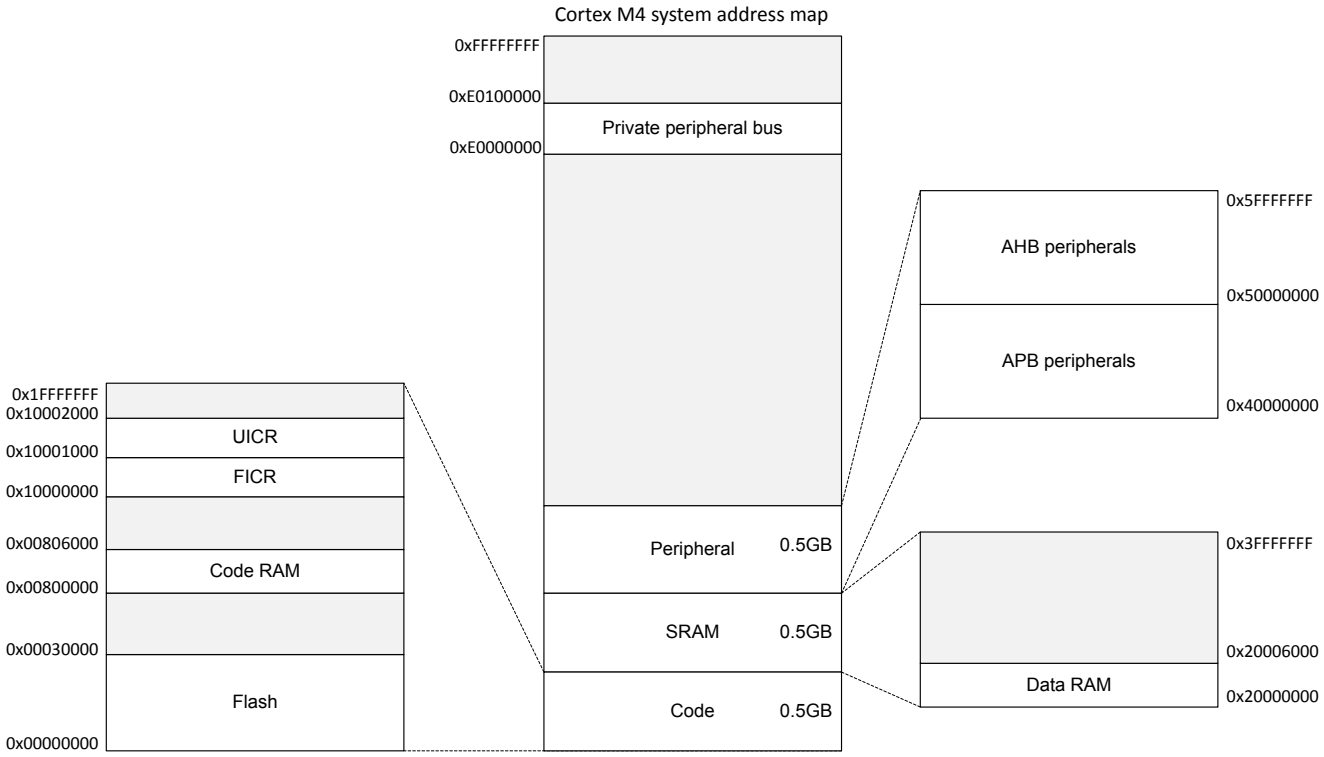



*Figure 3: Memory map*

## 4.2.4 Instantiation

| ID | Base address | Peripheral | Instance | Description | |
|---|---|---|---|---|---|
| 0 | 0x40000000 | BPROT | BPROT | Block protect | |
| 0 | 0x40000000 | CLOCK | CLOCK | Clock control | |
| 0 | 0x40000000 | POWER | POWER | Power control | |
| 0 | 0x50000000 | GPIO | P0 | General purpose input and output | |
| 1 | 0x40001000 | RADIO | RADIO | 2.4 GHz radio | |
| 2 | 0x40002000 | UART | UART0 | Universal asynchronous receiver/transmitter | Deprecated |
| 2 | 0x40002000 | UARTE | UARTE0 | Universal asynchronous receiver/transmitter with EasyDMA | |
| 3 | 0x40003000 | SPI | SPI1 | SPI master 1 | Deprecated |
| 3 | 0x40003000 | SPIM | SPIM1 | SPI master 1 | |
| 3 | 0x40003000 | SPIS | SPIS1 | SPI slave 1 | |
| 3 | 0x40003000 | TWI | TWI0 | Two-wire interface master | Deprecated |
| 3 | 0x40003000 | TWIM | TWIM0 | Two-wire interface master | |
| 3 | 0x40003000 | TWIS | TWIS0 | Two-wire interface slave | |
| 4 | 0x40004000 | SPI | SPI0 | SPI master 0 | Deprecated |
| 4 | 0x40004000 | SPIM | SPIM0 | SPI master 0 | |
| 4 | 0x40004000 | SPIS | SPIS0 | SPI slave 0 | |
| 6 | 0x40006000 | GPIOTE | GPIOTE | GPIO tasks and events | |
| 7 | 0x40007000 | SAADC | SAADC | Analog-to-digital converter | |
| 8 | 0x40008000 | TIMER | TIMER0 | Timer 0 | |
| 9 | 0x40009000 | TIMER | TIMER1 | Timer 1 | |
| 10 | 0x4000A000 | TIMER | TIMER2 | Timer 2 | |
| 11 | 0x4000B000 | RTC | RTC0 | Real-time counter 0 | |
| 12 | 0x4000C000 | TEMP | TEMP | Temperature sensor | |
| 13 | 0x4000D000 | RNG | RNG | Random number generator | |
| 14 | 0x4000E000 | ECB | ECB | AES Electronic Codebook (ECB) mode block encryption | |
| 15 | 0x4000F000 | AAR | AAR | Accelerated address resolver | |

| ID | Base address | Peripheral | Instance | Description |
|---|---|---|---|---|
| 15 | 0x4000F000 | CCM | CCM | AES CCM mode encryption |
| 16 | 0x40010000 | WDT | WDT | Watchdog timer |
| 17 | 0x40011000 | RTC | RTC1 | Real-time counter 1 |
| 18 | 0x40012000 | QDEC | QDEC | Quadrature decoder |
| 19 | 0x40013000 | COMP | COMP | General purpose comparator |
| 20 | 0x40014000 | EGU | EGU0 | Event generator unit 0 |
| 20 | 0x40014000 | SWI | SWI0 | Software interrupt 0 |
| 21 | 0x40015000 | EGU | EGU1 | Event generator unit 1 |
| 21 | 0x40015000 | SWI | SWI1 | Software interrupt 1 |
| 22 | 0x40016000 | SWI | SWI2 | Software interrupt 2 |
| 23 | 0x40017000 | SWI | SWI3 | Software interrupt 3 |
| 24 | 0x40018000 | SWI | SWI4 | Software interrupt 4 |
| 25 | 0x40019000 | SWI | SWI5 | Software interrupt 5 |
| 28 | 0x4001C000 | PWM | PWM0 | Pulse-width modulation unit 0 |
| 29 | 0x4001D000 | PDM | PDM | Pulse-density modulation (digital microphone interface) |
| 30 | 0x4001E000 | NVMC | NVMC | Non-volatile memory controller |
| 31 | 0x4001F000 | PPI | PPI | Programmable peripheral interconnect |
| N/A | 0x10000000 | FICR | FICR | Factory information configuration |
| N/A | 0x10001000 | UICR | UICR | User information configuration |

*Table 4: Instantiation table*

# 4.3 NVMC — Non-volatile memory controller

The non-volatile memory controller (NVMC) is used for writing and erasing of the internal flash memory and the UICR (user information configuration registers).

The CONFIG on page 21 is used to enable the NVMC for writing (CONFIG.WEN) and erasing (CONFIG.EEN). The user must make sure that writing and erasing are not enabled at the same time. Having both enabled at the same time may result in unpredictable behavior.

The CPU must be halted before initiating a NVMC operation from the debug system.

## 4.3.1 Writing to flash

When writing is enabled, full 32-bit words are written to word-aligned addresses in flash.

As illustrated in Memory on page 16, the flash is divided into multiple pages. The same 32-bit word in the flash can only be written $n_{WRITE}$ number of times before a page erase must be performed.

The NVMC is only able to write `0` to bits in the flash that are erased (set to `1`). It cannot rewrite a bit back to `1`. Only full 32-bit words can be written to flash using the NVMC interface. To write less than 32 bits, write the data as a full 32-bit word and set all the bits that should remain unchanged in the word to `1`. Note that the restriction on the number of writes ($n_{WRITE}$) still applies in this case.

Only word-aligned writes are allowed. Byte or half-word-aligned writes will result in a hard fault.

The time it takes to write a word to flash is specified by $t_{WRITE}$. The CPU is halted while the NVMC is writing to the flash.

## 4.3.2 Erasing a page in flash

When erase is enabled, the flash memory can be erased page by page using the ERASEPAGE on page 21.

After erasing a flash page, all bits in the page are set to 1. The time it takes to erase a page is specified by $t_{ERASEPAGE}$. The CPU is halted if the CPU executes code from the flash while the NVMC is writing to the flash.

See Partial erase of a page in flash on page 20 for information on dividing the page erase time into shorter chunks.

### 4.3.3 Writing to user information configuration registers (UICR)

User information configuration registers (UICR) are written in the same way as flash. After UICR has been written, the new UICR configuration will take effect after a reset.

UICR can only be written $n_{WRITE}$ number of times before an erase must be performed using ERASEUICR on page 23 or ERASEALL on page 22. The time it takes to write a word to UICR is specified by $t_{WRITE}$.
The CPU is halted while the NVMC is writing to the UICR.

### 4.3.4 Erasing user information configuration registers (UICR)

When erase is enabled, UICR can be erased using the ERASEUICR on page 23.

After erasing UICR all bits in UICR are set to 1. The time it takes to erase UICR is specified by $t_{ERASEPAGE}$. The CPU is halted if the CPU executes code from the flash while the NVMC performs the erase operation.

### 4.3.5 Erase all

When erase is enabled, flash and UICR can be erased completely in one operation by using ERASEALL on page 22. This operation will not erase the factory information configuration registers (FICR).

The time it takes to perform an ERASEALL command is specified by $t_{ERASEALL}$. The CPU is halted if the CPU executes code from the flash while the NVMC performs the erase operation.

### 4.3.6 Partial erase of a page in flash

Partial erase is a feature in the NVMC to split a page erase time into shorter chunks, so this can be used to prevent longer CPU stalls in time-critical applications. Partial erase is only applicable to the code area in the flash and does not work with UICR.

When erase is enabled, the partial erase of a flash page can be started by writing to ERASEPAGEPARTIAL on page 23. The duration of a partial erase can be configured in ERASEPAGEPARTIALCFG on page 23. A flash page is erased when its erase time reaches $t_{ERASEPAGE}$. Use ERASEPAGEPARTIAL N number of times so that N * ERASEPAGEPARTIALCFG ≥ $t_{ERASEPAGE}$, where N * ERASEPAGEPARTIALCFG gives the cumulative (total) erase time. Every time the cumulative erase time reaches $t_{ERASEPAGE}$, it counts as one erase cycle.

After the erase is done, all bits in the page are set to '1'. The CPU is halted if the CPU executes code from the flash while the NVMC performs the partial erase operation.

The bits in the page are undefined if the flash page erase is incomplete, i.e. if a partial erase has started but the total erase time is less than $t_{ERASEPAGE}$.

### 4.3.7 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x4001E000 | NVMC | NVMC | Non-volatile memory controller | |

*Table 5: Instances*

| Register | Offset | Description | |
|---|---|---|---|
| READY | 0x400 | Ready flag | |
| CONFIG | 0x504 | Configuration register | |
| ERASEPAGE | 0x508 | Register for erasing a page in code area | |
| ERASEPCR1 | 0x508 | Register for erasing a page in code area. Equivalent to ERASEPAGE. | Deprecated |
| ERASEALL | 0x50C | Register for erasing all non-volatile user memory | |
| ERASEPCR0 | 0x510 | Register for erasing a page in code area. Equivalent to ERASEPAGE. | Deprecated |
| ERASEUICR | 0x514 | Register for erasing user information configuration registers | |
| ERASEPAGEPARTIAL | 0x518 | Register for partial erase of a page in code area | |
| ERASEPAGEPARTIALCFG | 0x51C | Register for partial erase configuration | |

*Table 6: Register overview*

### 4.3.7.1 READY

Address offset: 0x400

Ready flag

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000001** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | READY | | | NVMC is ready or busy |
| | | | Busy | 0 | NVMC is busy (ongoing write or erase operation) |
| | | | Ready | 1 | NVMC is ready |

### 4.3.7.2 CONFIG

Address offset: 0x504

Configuration register

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | WEN | | | Program memory access mode. It is strongly recommended to activate erase and write modes only when they are actively used. |
| | | | Ren | 0 | Read only access |
| | | | Wen | 1 | Write enabled |
| | | | Een | 2 | Erase enabled |

### 4.3.7.3 ERASEPAGE

Address offset: 0x508

Register for erasing a page in code area

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ERASEPAGE | | | Register for starting erase of a page in code area.<br>The value is the address to the page to be erased (addresses of first word in page). Note that the erase must be enabled using CONFIG.WEN before the page can be erased.<br>Attempts to erase pages that are outside the code area may result in undesirable behavior, e.g. the wrong page may be erased. |

### 4.3.7.4 ERASEPCR1 ( Deprecated )

Address offset: 0x508

Register for erasing a page in code area. Equivalent to ERASEPAGE.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ERASEPCR1 | | | Register for erasing a page in code area. Equivalent to ERASEPAGE. |

### 4.3.7.5 ERASEALL

Address offset: 0x50C

Register for erasing all non-volatile user memory

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ERASEALL | | | Erase all non-volatile memory including UICR registers. Note that the erase must be enabled using CONFIG.WEN before the non-volatile memory can be erased. |
| | | | NoOperation | 0 | No operation |
| | | | Erase | 1 | Start erase of chip |

### 4.3.7.6 ERASEPCR0 ( Deprecated )

Address offset: 0x510

Register for erasing a page in code area. Equivalent to ERASEPAGE.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ERASEPCR0 | | | Register for starting erase of a page in code area. Equivalent to ERASEPAGE. |

### 4.3.7.7 ERASEUICR

Address offset: 0x514

Register for erasing user information configuration registers

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ERASEUICR | | | Register starting erase of all user information configuration registers. Note that the erase must be enabled using CONFIG.WEN before the UICR can be erased. |
| | | | NoOperation | 0 | No operation |
| | | | Erase | 1 | Start erase of UICR |

### 4.3.7.8 ERASEPAGEPARTIAL

Address offset: 0x518

Register for partial erase of a page in code area

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ERASEPAGEPARTIAL | | | Register for starting partial erase of a page in code area<br>The value is the address to the page to be partially erased (address of the first word in page). Note that the erase must be enabled using CONFIG.WEN before every erase page partial and disabled using CONFIG.WEN after every erase page partial. Attempts to erase pages that are outside the code area may result in undesirable behaviour, e.g. the wrong page may be erased. |

### 4.3.7.9 ERASEPAGEPARTIALCFG

Address offset: 0x51C

Register for partial erase configuration

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A |
| **Reset 0x0000000A** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | DURATION | | | Duration of the partial erase in milliseconds<br>The user must ensure that the total erase time is long enough for a complete erase of the flash page. |

### 4.3.8 Electrical specification

#### 4.3.8.1 Flash programming

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $n_{WRITE}$ | Number of times a 32-bit word can be written before erase | | | 2 | |
| $n_{ENDURANCE}$ | Erase cycles per page | 10000 | | | |
| $t_{WRITE}$ | Time to write one 32-bit word | | | 41[2] | µs |
| $t_{ERASEPAGE}$ | Time to erase one page | | | 85[2] | ms |
| $t_{ERASEALL}$ | Time to erase all flash | | | 169[2] | ms |
| $t_{ERASEPAGEPARTIAL,acc}$ | Accuracy of the partial page erase duration. Total execution time for one partial page erase is defined as ERASEPAGEPARTIALCFG * $t_{ERASEPAGEPARTIAL,acc}$. | | | 1.05[2] | |

## 4.4 FICR — Factory information configuration registers

Factory information configuration registers (FICR) are pre-programmed in factory and cannot be erased by the user. These registers contain chip-specific information and configuration.

### 4.4.1 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x10000000 | FICR | FICR | Factory information configuration | |

*Table 7: Instances*

| Register | Offset | Description |
|---|---|---|
| CODEPAGESIZE | 0x010 | Code memory page size |
| CODESIZE | 0x014 | Code memory size |
| DEVICEID[0] | 0x060 | Device identifier |
| DEVICEID[1] | 0x064 | Device identifier |
| ER[0] | 0x080 | Encryption root, word 0 |
| ER[1] | 0x084 | Encryption root, word 1 |
| ER[2] | 0x088 | Encryption root, word 2 |
| ER[3] | 0x08C | Encryption root, word 3 |
| IR[0] | 0x090 | Identity root, word 0 |
| IR[1] | 0x094 | Identity root, word 1 |
| IR[2] | 0x098 | Identity root, word 2 |
| IR[3] | 0x09C | Identity root, word 3 |
| DEVICEADDRTYPE | 0x0A0 | Device address type |
| DEVICEADDR[0] | 0x0A4 | Device address 0 |
| DEVICEADDR[1] | 0x0A8 | Device address 1 |
| INFO.PART | 0x100 | Part code |
| INFO.VARIANT | 0x104 | Part variant, hardware version and production configuration |
| INFO.PACKAGE | 0x108 | Package option |
| INFO.RAM | 0x10C | RAM variant |
| INFO.FLASH | 0x110 | Flash variant |
| INFO.UNUSED8[0] | 0x114 | Reserved |

[2] HFXO is used here

| Register | Offset | Description | |
|---|---|---|---|
| INFO.UNUSED8[1] | 0x118 | | Reserved |
| INFO.UNUSED8[2] | 0x11C | | Reserved |
| TEMP.A0 | 0x404 | Slope definition A0 | |
| TEMP.A1 | 0x408 | Slope definition A1 | |
| TEMP.A2 | 0x40C | Slope definition A2 | |
| TEMP.A3 | 0x410 | Slope definition A3 | |
| TEMP.A4 | 0x414 | Slope definition A4 | |
| TEMP.A5 | 0x418 | Slope definition A5 | |
| TEMP.B0 | 0x41C | Y-intercept B0 | |
| TEMP.B1 | 0x420 | Y-intercept B1 | |
| TEMP.B2 | 0x424 | Y-intercept B2 | |
| TEMP.B3 | 0x428 | Y-intercept B3 | |
| TEMP.B4 | 0x42C | Y-intercept B4 | |
| TEMP.B5 | 0x430 | Y-intercept B5 | |
| TEMP.T0 | 0x434 | Segment end T0 | |
| TEMP.T1 | 0x438 | Segment end T1 | |
| TEMP.T2 | 0x43C | Segment end T2 | |
| TEMP.T3 | 0x440 | Segment end T3 | |
| TEMP.T4 | 0x444 | Segment end T4 | |

*Table 8: Register overview*

### 4.4.1.1 CODEPAGESIZE

Address offset: 0x010

Code memory page size

| Bit number | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| **Reset 0x00001000** | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ID | Acce | Field | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| A | R | CODEPAGESIZE | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Value ID

Value

Description

Code memory page size

### 4.4.1.2 CODESIZE

Address offset: 0x014

Code memory size

| Bit number | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| **Reset 0x00000030** | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ID | Acce | Field | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| A | R | CODESIZE | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Value ID

Value

Description

Code memory size in number of pages

Total code space is: CODEPAGESIZE * CODESIZE

### 4.4.1.3 DEVICEID[n] (n=0..1)

Address offset: 0x060 + (n × 0x4)

Device identifier

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | DEVICEID | | | 64 bit unique device identifier<br><br>DEVICEID[0] contains the least significant bits of the device identifier. DEVICEID[1] contains the most significant bits of the device identifier. |

### 4.4.1.4 ER[n] (n=0..3)

Address offset: 0x080 + (n × 0x4)

Encryption root, word n

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | ER | | | Encryption root, word n |

### 4.4.1.5 IR[n] (n=0..3)

Address offset: 0x090 + (n × 0x4)

Identity root, word n

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | IR | | | Identity root, word n |

### 4.4.1.6 DEVICEADDRTYPE

Address offset: 0x0A0

Device address type

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | DEVICEADDRTYPE | | | Device address type |
| | | | Public | 0 | Public address |
| | | | Random | 1 | Random address |

### 4.4.1.7 DEVICEADDR[n] (n=0..1)

Address offset: 0x0A4 + (n × 0x4)

Device address n

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | DEVICEADDR | | | 48 bit device address<br>DEVICEADDR[0] contains the least significant bits of the device address. DEVICEADDR[1] contains the most significant bits of the device address. Only bits [15:0] of DEVICEADDR[1] are used. |

### 4.4.1.8 INFO.PART

Address offset: 0x100

Part code

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00052811** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 1 0 1 0 0 0 0 0 0 1 0 0 0 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | PART | | | Part code |
| | | | N52810 | 0x52810 | nRF52810 |
| | | | N52811 | 0x52811 | nRF52811 |
| | | | Unspecified | 0xFFFFFFFF | Unspecified |

### 4.4.1.9 INFO.VARIANT

Address offset: 0x104

Part variant, hardware version and production configuration

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | VARIANT | | | Part variant, hardware version and production configuration, encoded as ASCII |
| | | | AAAA | 0x41414141 | AAAA |
| | | | AAA0 | 0x41414130 | AAA0 |
| | | | AABA | 0x41414241 | AABA |
| | | | AABB | 0x41414242 | AABB |
| | | | AAB0 | 0x41414230 | AAB0 |
| | | | AACA | 0x41414341 | AACA |
| | | | AACB | 0x41414342 | AACB |
| | | | AAC0 | 0x41414330 | AAC0 |
| | | | Unspecified | 0xFFFFFFFF | Unspecified |

### 4.4.1.10 INFO.PACKAGE

Address offset: 0x108

Package option

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | PACKAGE | | | Package option |
| | | | QF | 0x2000 | QFxx - 48-pin QFN |
| | | | QC | 0x2003 | QCxx - 32-pin QFN |
| | | | CA | 0x2004 | CAxx - WLCSP |
| | | | Unspecified | 0xFFFFFFFF | Unspecified |

### 4.4.1.11 INFO.RAM

Address offset: 0x10C

RAM variant

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000018** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | RAM | | | RAM variant |
| | | | K24 | 0x18 | 24 kByte RAM |
| | | | Unspecified | 0xFFFFFFFF | Unspecified |

### 4.4.1.12 INFO.FLASH

Address offset: 0x110

Flash variant

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x000000C0** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | FLASH | | | Flash variant |
| | | | K192 | 0xC0 | 192 kByte flash |
| | | | Unspecified | 0xFFFFFFFF | Unspecified |

### 4.4.1.13 TEMP.A0

Address offset: 0x404

Slope definition A0

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A |
| **Reset 0x00000320** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | A | | | A (slope definition) register |

### 4.4.1.14 TEMP.A1

Address offset: 0x408

Slope definition A1

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000343** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 0 0 1 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | A | | | A (slope definition) register |

### 4.4.1.15 TEMP.A2

Address offset: 0x40C

Slope definition A2

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x0000035D** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 1 1 1 0 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | A | | | A (slope definition) register |

### 4.4.1.16 TEMP.A3

Address offset: 0x410

Slope definition A3

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000400** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | A | | | A (slope definition) register |

### 4.4.1.17 TEMP.A4

Address offset: 0x414

Slope definition A4

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000452** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 1 0 0 1 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | A | | | A (slope definition) register |

### 4.4.1.18 TEMP.A5

Address offset: 0x418

Slope definition A5

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x0000037B** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 0 1 1 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | A | | | A (slope definition) register |

### 4.4.1.19 TEMP.B0

Address offset: 0x41C

Y-intercept B0

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00003FCC** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 1 1 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | B | | | B (y-intercept) |

### 4.4.1.20 TEMP.B1

Address offset: 0x420

Y-intercept B1

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00003F98** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0 1 1 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | B | | | B (y-intercept) |

### 4.4.1.21 TEMP.B2

Address offset: 0x424

Y-intercept B2

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00003F98** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0 1 1 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | B | | | B (y-intercept) |

### 4.4.1.22 TEMP.B3

Address offset: 0x428

Y-intercept B3

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000012** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | B | | | B (y-intercept) |

### 4.4.1.23 TEMP.B4

Address offset: 0x42C

Y-intercept B4

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A |
| **Reset 0x0000004D** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 0 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | B | | | B (y-intercept) |

## 4.4.1.24 TEMP.B5

Address offset: 0x430

Y-intercept B5

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A |
| **Reset 0x00003E10** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 0 1 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | B | | | B (y-intercept) |

## 4.4.1.25 TEMP.T0

Address offset: 0x434

Segment end T0

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A |
| **Reset 0x000000E2** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 1 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | T | | | T (segment end) register |

## 4.4.1.26 TEMP.T1

Address offset: 0x438

Segment end T1

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | T | | | T (segment end) register |

## 4.4.1.27 TEMP.T2

Address offset: 0x43C

Segment end T2

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A |
| **Reset 0x00000014** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | T | | | T (segment end) register |

### 4.4.1.28 TEMP.T3

Address offset: 0x440

Segment end T3

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A |
| **Reset 0x00000019** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | T | | | T (segment end) register |

### 4.4.1.29 TEMP.T4

Address offset: 0x444

Segment end T4

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A |
| **Reset 0x00000050** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | T | | | T (segment end) register |

## 4.5 UICR — User information configuration registers

The user information configuration registers (UICRs) are non-volatile memory (NVM) registers for configuring user-specific settings.

For information on writing UICR registers, see the NVMC — Non-volatile memory controller on page 19 and Memory on page 16 chapters.

### 4.5.1 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x10001000 | UICR | UICR | User information configuration | |

*Table 9: Instances*

| Register | Offset | Description | |
|---|---|---|---|
| UNUSED0 | 0x000 | | Reserved |
| UNUSED1 | 0x004 | | Reserved |
| UNUSED2 | 0x008 | | Reserved |
| UNUSED3 | 0x010 | | Reserved |
| NRFFW[0] | 0x014 | Reserved for Nordic firmware design | |
| NRFFW[1] | 0x018 | Reserved for Nordic firmware design | |
| NRFFW[2] | 0x01C | Reserved for Nordic firmware design | |
| NRFFW[3] | 0x020 | Reserved for Nordic firmware design | |
| NRFFW[4] | 0x024 | Reserved for Nordic firmware design | |
| NRFFW[5] | 0x028 | Reserved for Nordic firmware design | |
| NRFFW[6] | 0x02C | Reserved for Nordic firmware design | |
| NRFFW[7] | 0x030 | Reserved for Nordic firmware design | |

| Register | Offset | Description |
|---|---|---|
| NRFFW[8] | 0x034 | Reserved for Nordic firmware design |
| NRFFW[9] | 0x038 | Reserved for Nordic firmware design |
| NRFFW[10] | 0x03C | Reserved for Nordic firmware design |
| NRFFW[11] | 0x040 | Reserved for Nordic firmware design |
| NRFFW[12] | 0x044 | Reserved for Nordic firmware design |
| NRFHW[0] | 0x050 | Reserved for Nordic hardware design |
| NRFHW[1] | 0x054 | Reserved for Nordic hardware design |
| NRFHW[2] | 0x058 | Reserved for Nordic hardware design |
| NRFHW[3] | 0x05C | Reserved for Nordic hardware design |
| NRFHW[4] | 0x060 | Reserved for Nordic hardware design |
| NRFHW[5] | 0x064 | Reserved for Nordic hardware design |
| NRFHW[6] | 0x068 | Reserved for Nordic hardware design |
| NRFHW[7] | 0x06C | Reserved for Nordic hardware design |
| NRFHW[8] | 0x070 | Reserved for Nordic hardware design |
| NRFHW[9] | 0x074 | Reserved for Nordic hardware design |
| NRFHW[10] | 0x078 | Reserved for Nordic hardware design |
| NRFHW[11] | 0x07C | Reserved for Nordic hardware design |
| CUSTOMER[0] | 0x080 | Reserved for customer |
| CUSTOMER[1] | 0x084 | Reserved for customer |
| CUSTOMER[2] | 0x088 | Reserved for customer |
| CUSTOMER[3] | 0x08C | Reserved for customer |
| CUSTOMER[4] | 0x090 | Reserved for customer |
| CUSTOMER[5] | 0x094 | Reserved for customer |
| CUSTOMER[6] | 0x098 | Reserved for customer |
| CUSTOMER[7] | 0x09C | Reserved for customer |
| CUSTOMER[8] | 0x0A0 | Reserved for customer |
| CUSTOMER[9] | 0x0A4 | Reserved for customer |
| CUSTOMER[10] | 0x0A8 | Reserved for customer |
| CUSTOMER[11] | 0x0AC | Reserved for customer |
| CUSTOMER[12] | 0x0B0 | Reserved for customer |
| CUSTOMER[13] | 0x0B4 | Reserved for customer |
| CUSTOMER[14] | 0x0B8 | Reserved for customer |
| CUSTOMER[15] | 0x0BC | Reserved for customer |
| CUSTOMER[16] | 0x0C0 | Reserved for customer |
| CUSTOMER[17] | 0x0C4 | Reserved for customer |
| CUSTOMER[18] | 0x0C8 | Reserved for customer |
| CUSTOMER[19] | 0x0CC | Reserved for customer |
| CUSTOMER[20] | 0x0D0 | Reserved for customer |
| CUSTOMER[21] | 0x0D4 | Reserved for customer |
| CUSTOMER[22] | 0x0D8 | Reserved for customer |
| CUSTOMER[23] | 0x0DC | Reserved for customer |
| CUSTOMER[24] | 0x0E0 | Reserved for customer |
| CUSTOMER[25] | 0x0E4 | Reserved for customer |
| CUSTOMER[26] | 0x0E8 | Reserved for customer |
| CUSTOMER[27] | 0x0EC | Reserved for customer |
| CUSTOMER[28] | 0x0F0 | Reserved for customer |
| CUSTOMER[29] | 0x0F4 | Reserved for customer |
| CUSTOMER[30] | 0x0F8 | Reserved for customer |
| CUSTOMER[31] | 0x0FC | Reserved for customer |
| PSELRESET[0] | 0x200 | Mapping of the nRESET function (see POWER chapter for details) |
| PSELRESET[1] | 0x204 | Mapping of the nRESET function (see POWER chapter for details) |
| APPROTECT | 0x208 | Access port protection |

*Table 10: Register overview*

### 4.5.1.1 NRFFW[n] (n=0..12)

Address offset: 0x014 + (n × 0x4)

Reserved for Nordic firmware design

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0xFFFFFFFF** | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | NRFFW | | | Reserved for Nordic firmware design |

### 4.5.1.2 NRFHW[n] (n=0..11)

Address offset: 0x050 + (n × 0x4)

Reserved for Nordic hardware design

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0xFFFFFFFF** | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | NRFHW | | | Reserved for Nordic hardware design |

### 4.5.1.3 CUSTOMER[n] (n=0..31)

Address offset: 0x080 + (n × 0x4)

Reserved for customer

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0xFFFFFFFF** | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | CUSTOMER | | | Reserved for customer |

### 4.5.1.4 PSELRESET[n] (n=0..1)

Address offset: 0x200 + (n × 0x4)

Mapping of the nRESET function (see POWER chapter for details)

All PSELRESET registers have to contain the same value for a pin mapping to be valid. If values are not the same, there will be no nRESET function exposed on a GPIO. As a result, the device will always start independently of the levels present on any of the GPIOs.

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | 21 | GPIO pin number onto which nRESET is exposed |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

### 4.5.1.5 APPROTECT

Address offset: 0x208

Access port protection

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce Field | Value ID | Value | Description |
| A | RW PALL | | | Enable or disable access port protection.<br><br>See Debug on page 38 for more information. |
| | | Disabled | 0xFF | Disable |
| | | Enabled | 0x00 | Enable |

# 4.6 EasyDMA

EasyDMA is a module implemented by some peripherals to gain direct access to Data RAM.

EasyDMA is an AHB bus master similar to CPU and is connected to the AHB multilayer interconnect for direct access to Data RAM. EasyDMA is not able to access flash.

A peripheral can implement multiple EasyDMA instances to provide dedicated channels. For example, for reading and writing of data between the peripheral and RAM. This concept is illustrated in EasyDMA example on page 35.

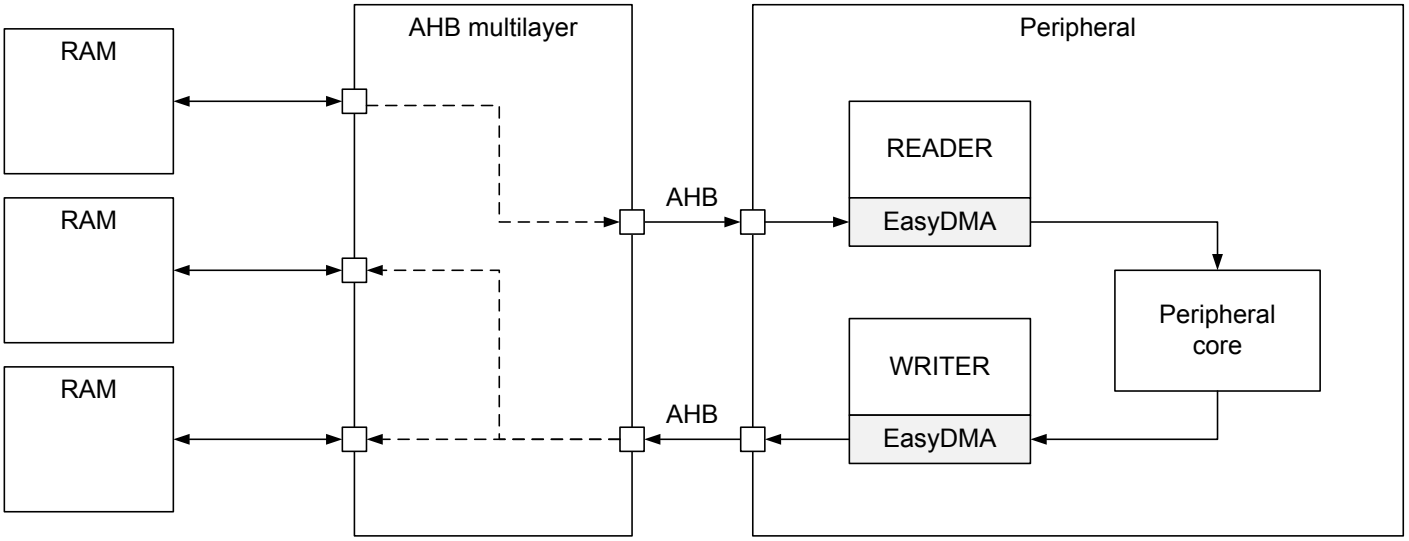



*Figure 4: EasyDMA example*

An EasyDMA channel is implemented in the following way, but some variations may occur:

```
READERBUFFER_SIZE 5
WRITERBUFFER_SIZE 6

uint8_t readerBuffer[READERBUFFER_SIZE]  __at__ 0x20000000;
uint8_t writerBuffer[WRITERBUFFER_SIZE]  __at__ 0x20000005;

// Configuring the READER channel
MYPERIPHERAL->READER.MAXCNT = READERBUFFER_SIZE;
MYPERIPHERAL->READER.PTR = &readerBuffer;

// Configure the WRITER channel
MYPERIPHERAL->WRITER.MAXCNT = WRITEERBUFFER_SIZE;
MYPERIPHERAL->WRITER.PTR = &writerBuffer;
```

This example shows a peripheral called MYPERIPHERAL that implements two EasyDMA channels - one for reading called READER, and one for writing called WRITER. When the peripheral is started, it is assumed that the peripheral will:

- Read 5 bytes from the readerBuffer located in RAM at address 0x20000000.
- Process the data.
- Write no more than 6 bytes back to the writerBuffer located in RAM at address 0x20000005.

The memory layout of these buffers is illustrated in EasyDMA memory layout on page 36.

| | | | | |
|---|---|---|---|---|
| 0x20000000 | readerBuffer[0] | readerBuffer[1] | readerBuffer[2] | readerBuffer[3] |
| 0x20000004 | readerBuffer[4] | writerBuffer[0] | writerBuffer[1] | writerBuffer[2] |
| 0x20000008 | writerBuffer[3] | writerBuffer[4] | writerBuffer[5] | |

*Figure 5: EasyDMA memory layout*

The WRITER.MAXCNT register should not be specified larger than the actual size of the buffer (writerBuffer). Otherwise, the channel would overflow the writerBuffer.

Once an EasyDMA transfer is completed, the AMOUNT register can be read by the CPU to see how many bytes were transferred. For example, CPU can read MYPERIPHERAL->WRITER.AMOUNT register to see how many bytes WRITER wrote to RAM.

Note that the PTR register of a READER or WRITER must point to a valid memory region before use. The reset value of a PTR register is not guaranteed to point to valid memory. See Memory on page 16 for more information about the different memory regions and EasyDMA connectivity.

### 4.6.1 EasyDMA error handling

Some errors may occur during DMA handling.

If READER.PTR or WRITER.PTR is not pointing to a valid memory region, an EasyDMA transfer may result in a HardFault or RAM corruption. See Memory on page 16 for more information about the different memory regions.

If several AHB bus masters try to access the same AHB slave at the same time, AHB bus congestion might occur. An EasyDMA channel is an AHB master. Depending on the peripheral, the peripheral may either stall and wait for access to be granted, or lose data.

### 4.6.2 EasyDMA array list

EasyDMA is able to operate in Array List mode.

The Array List mode is implemented in channels where the LIST register is available.

The array list does not provide a mechanism to explicitly specify where the next item in the list is located. Instead, it assumes that the list is organized as a linear array where items are located one after the other in RAM.

The EasyDMA Array List can be implemented by using the data structure ArrayList_type as illustrated in the code example below using a READER EasyDMA channel as an example:

```
#define BUFFER_SIZE  4

typedef struct ArrayList
{
  uint8_t buffer[BUFFER_SIZE];
} ArrayList_type;

ArrayList_type ReaderList[3]  __at__ 0x20000000;

MYPERIPHERAL->READER.MAXCNT = BUFFER_SIZE;
MYPERIPHERAL->READER.PTR = &ReaderList;
MYPERIPHERAL->READER.LIST = MYPERIPHERAL_READER_LIST_ArrayList;
```

The data structure only includes a buffer with size equal to the size of READER.MAXCNT register. EasyDMA uses the READER.MAXCNT register to determine when the buffer is full.

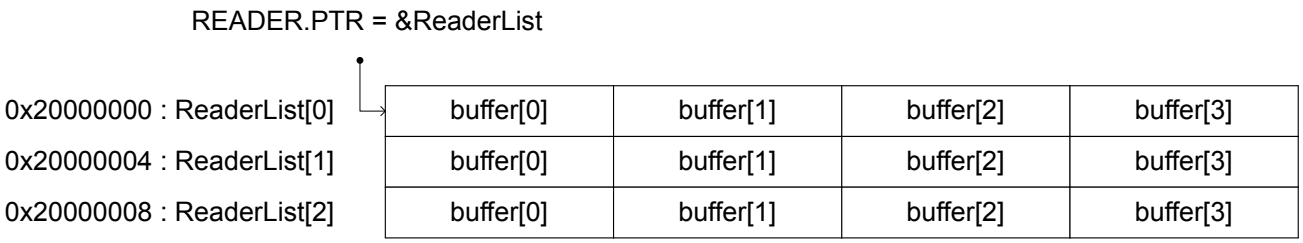



*Figure 6: EasyDMA array list*

## 4.7 AHB multilayer

AHB multilayer enables parallel access paths between multiple masters and slaves in a system. Access is resolved using priorities.

Each bus master is connected to the slave devices using an interconnection matrix. The bus masters are assigned priorities. Priorities are used to resolve access when two (or more) bus masters request access to the same slave device. The following applies:

- If two (or more) bus masters request access to the same slave device, the master with the highest priority is granted the access first.
- Bus masters with lower priority are stalled until the higher priority master has completed its transaction.
- If the higher priority master pauses at any point during its transaction, the lower priority master in queue is temporarily granted access to the slave device until the higher priority master resumes its activity.
- Bus masters that have the same priority are mutually exclusive, thus cannot be used concurrently.

Below is a list of bus masters in the system and their priorities.

| Bus master name | Description |
|---|---|
| CPU | |
| SPIM0/SPIS0 | Same priority and mutually exclusive |
| RADIO | |
| CCM/ECB/AAR | Same priority and mutually exclusive |
| SAADC | |
| UARTE0 | |
| TWIM0/TWIS0 | Same priority and mutually exclusive |
| PDM | |
| PWM | |

*Table 11: AHB bus masters (listed in priority order, highest to lowest)*

Defined bus masters are the CPU and the peripherals with implemented EasyDMA, and the available slaves are RAM AHB slaves. How the bus masters and slaves are connected using the interconnection matrix is illustrated in Memory on page 16.

# 4.8 Debug

Debug system offers a flexible and powerful mechanism for non-intrusive debugging.

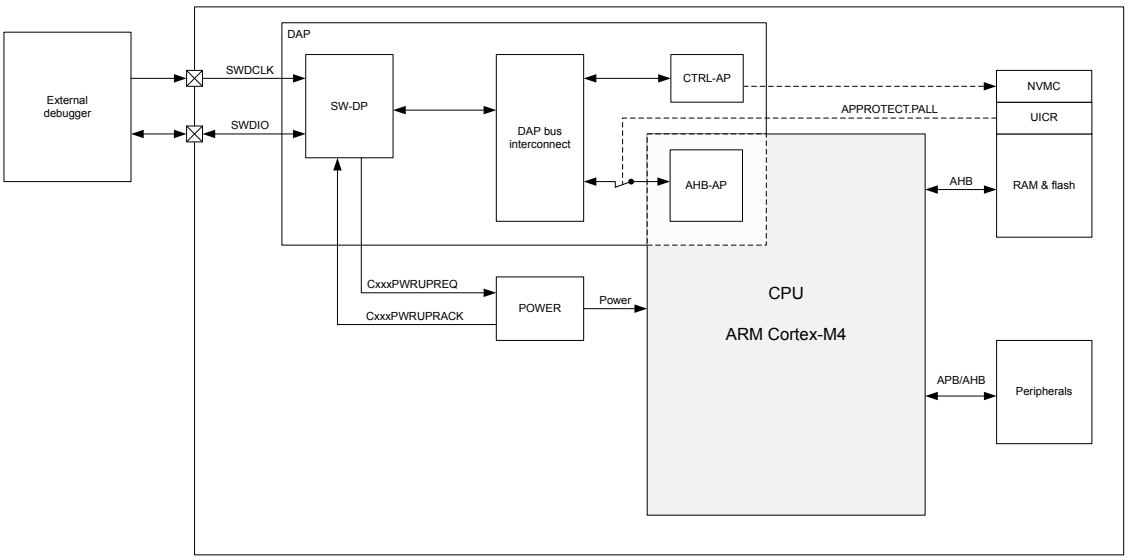



*Figure 7: Overview*

The main features of the debug system are:

- Two-pin serial wire debug (SWD) interface
- Flash patch and breakpoint (FPB) unit supports:

- Two literal comparators
- Six instruction comparators

## 4.8.1 DAP - Debug access port

An external debugger can access the device via the DAP.

The debug access port (DAP) implements a standard ARM® CoreSight™ serial wire debug port (SW-DP), which implements the serial wire debug protocol (SWD). SWD is a two-pin serial interface, see SWDCLK and SWDIO in Overview on page 38.

In addition to the default access port in CPU (AHB-AP), the DAP includes a custom control access port (CTRL-AP). The CTRL-AP is described in more detail in CTRL-AP - Control access port on page 39.

**Note:**

- The SWDIO line has an internal pull-up resistor.
- The SWDCLK line has an internal pull-down resistor.

## 4.8.2 CTRL-AP - Control access port

The control access port (CTRL-AP) is a custom access port that enables control of the device when other access ports in the DAP are disabled by the access port protection.

Access port protection blocks the debugger from read and write access to all CPU registers and memory-mapped addresses. See the UICR register APPROTECT on page 35 for more information on enabling access port protection.

Control access port has the following features:

- Soft reset, see Reset on page 52 for more information
- Disabling of access port protection, which is the reason why CTRL-AP allows control of the device even when all other access ports in the DAP are disabled by the access port protection

Access port protection is disabled by issuing an ERASEALL command via CTRL-AP. This command will erase the flash, UICR, and RAM.

### 4.8.2.1 Registers

| Register | Offset | Description |
|---|---|---|
| RESET | 0x000 | Soft reset triggered through CTRL-AP |
| ERASEALL | 0x004 | Erase all |
| ERASEALLSTATUS | 0x008 | Status register for the ERASEALL operation |
| APPROTECTSTATUS | 0x00C | Status register for access port protection |
| IDR | 0x0FC | CTRL-AP identification register, IDR |

*Table 12: Register overview*

#### 4.8.2.1.1 RESET

Address offset: 0x000

Soft reset triggered through CTRL-AP

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | RESET | | | Soft reset triggered through CTRL-AP. See Reset behavior in POWER chapter for more details. |
| | | | NoReset | 0 | Reset is not active |
| | | | Reset | 1 | Reset is active. Device is held in reset. |

### 4.8.2.1.2 ERASEALL

Address offset: 0x004

Erase all

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | ERASEALL | | | Erase all flash and RAM |
| | | | NoOperation | 0 | No operation |
| | | | Erase | 1 | Erase all flash and RAM |

### 4.8.2.1.3 ERASEALLSTATUS

Address offset: 0x008

Status register for the ERASEALL operation

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | ERASEALLSTATUS | | | Status register for the ERASEALL operation |
| | | | Ready | 0 | ERASEALL is ready |
| | | | Busy | 1 | ERASEALL is busy (on-going) |

### 4.8.2.1.4 APPROTECTSTATUS

Address offset: 0x00C

Status register for access port protection

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | APPROTECTSTATUS | | | Status register for access port protection |
| | | | Enabled | 0 | Access port protection enabled |
| | | | Disabled | 1 | Access port protection not enabled |

### 4.8.2.1.5 IDR

Address offset: 0x0FC

CTRL-AP identification register, IDR

| Bit number | | | | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | | | | E | E | E | E | D | D | D | D | C | C | C | C | C | C | C | B | B | B | B | | | | | | | A | A | A | A | A | A | A |
| **Reset 0x02880000** | | | | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | APID | | | AP identification |
| B | R | CLASS | | | Access port (AP) class |
| | | | NotDefined | 0x0 | No defined class |
| | | | MEMAP | 0x8 | Memory access port |
| C | R | JEP106ID | | | JEDEC JEP106 identity code |
| D | R | JEP106CONT | | | JEDEC JEP106 continuation code |
| E | R | REVISION | | | Revision |

### 4.8.2.2 Electrical specification

#### 4.8.2.2.1 Control access port

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $R_{pull}$ | Internal SWDIO and SWDCLK pull up/down resistance | | 13 | | kΩ |
| $f_{SWDCLK}$ | SWDCLK frequency | 0.125 | | 8 | MHz |

## 4.8.3 Debug interface mode

Before an external debugger can access either CPU's access port (AHB-AP) or the control access port (CTRL-AP), the debugger must first request the device to power up via CxxxPWRUPREQ in the SWJ-DP.

If the device is in System OFF when power is requested via CxxxPWRUPREQ, the system will wake up and the DIF flag in RESETREAS on page 57 will be set. The device is in the debug interface mode as long as the debugger is requesting power via CxxxPWRUPREQ. Once the debugger stops requesting power via CxxxPWRUPREQ, the device is back in normal mode. Some peripherals behave differently in Debug Interface mode compared to normal mode. These differences are described in more detail in the chapters of the peripherals that are affected.

When a debug session is over, the external debugger must make sure to put the device back into normal mode since the overall power consumption is higher in debug interface mode than in normal mode.

For details on how to use the debug capabilities, read the debug documentation of your IDE.

## 4.8.4 Real-time debug

The nRF52811 supports real-time debugging.

Real-time debugging allows interrupts to execute to completion in real time when breakpoints are set in thread mode or lower priority interrupts. This enables developers to set breakpoints and single-step through the code without the risk of real-time event-driven threads running at higher priority failing. For example, this enables the device to continue to service the high-priority interrupts of an external controller or sensor without failure or loss of state synchronization while the developer steps through code in a low-priority thread.

# 5 Power and clock management

## 5.1 Power management unit (PMU)

Power and clock management in nRF52811 is designed to automatically ensure maximum power efficiency.

The core of the power and clock management system is the power management unit (PMU) illustrated in Power management unit on page 42.

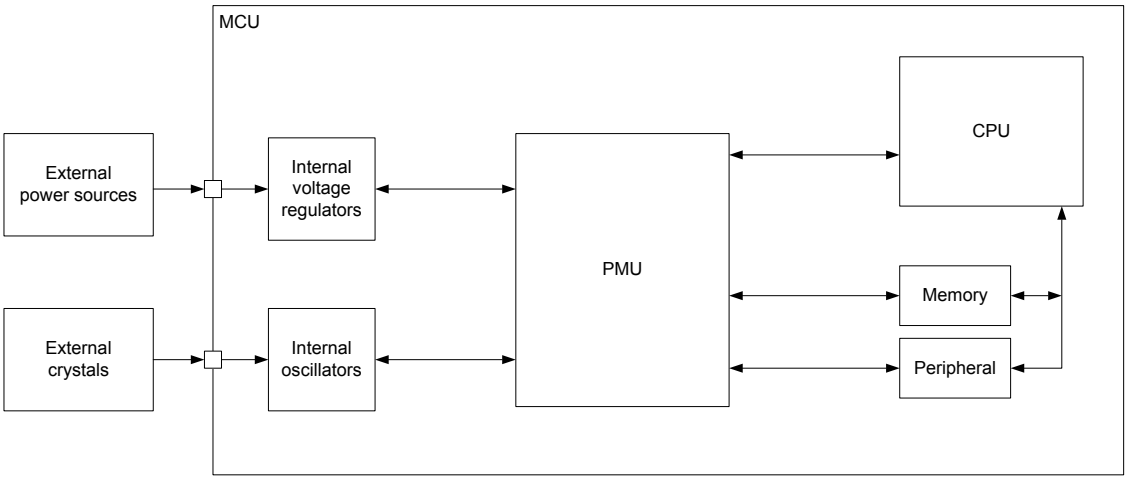



*Figure 8: Power management unit*

The PMU automatically detects which power and clock resources are required by the different components in the system at any given time. It will then start/stop and choose operation modes in supply regulators and clock sources, without user interaction, to achieve the lowest power consumption possible.

## 5.2 Current consumption

As the system is being constantly tuned by the Power management unit (PMU) on page 42, estimating the current consumption of an application can be challenging if the designer is not able to perform measurements directly on the hardware. To facilitate the estimation process, a set of current consumption scenarios are provided to show the typical current drawn from the VDD supply.

Each scenario specifies a set of operations and conditions applying to the given scenario. Current consumption scenarios, common conditions on page 43 shows a set of common conditions used in all scenarios, unless otherwise stated in the description of a given scenario. All scenarios are listed in Electrical specification on page 43.

| Condition | Value |
|---|---|
| VDD | 3 V |
| Temperature | 25°C |
| CPU | WFI (wait for interrupt)/WFE (wait for event) sleep |
| Peripherals | All idle |
| Clock | Not running |
| Regulator | LDO |
| RAM | Full 24 kB retention |
| Compiler[3] | GCC v4.9.3 20150529 (arm-none-eabi-gcc). Compiler flags: -O0 -falign-functions=16 -fno-strict-aliasing -mcpu=cortex-m4 -mfloat-abi=soft -msoft-float -mthumb. |
| 32 MHz crystal[4] | SMD 2520, 32 MHz, 10 pF +/- 10 ppm |

*Table 13: Current consumption scenarios, common conditions*

## 5.2.1 Electrical specification

### 5.2.1.1 CPU running

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $I_{CPU0}$ | CPU running CoreMark @64 MHz from flash, Clock = HFXO, Regulator = DCDC | | 2.2 | | mA |
| $I_{CPU1}$ | CPU running CoreMark @64 MHz from flash, Clock = HFXO | | 4.2 | | mA |
| $I_{CPU2}$ | CPU running CoreMark @64 MHz from RAM, Clock = HFXO, Regulator = DCDC | | 2.1 | | mA |
| $I_{CPU3}$ | CPU running CoreMark @64 MHz from RAM, Clock = HFXO | | 4 | | mA |
| $I_{CPU4}$ | CPU running CoreMark @64 MHz from flash, Clock = HFINT, Regulator = DCDC | | 2 | | mA |

[3] Applying only when CPU is running
[4] Applying only when HFXO is running

### 5.2.1.2 Radio transmitting/receiving

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $I_{RADIO\_TX0}$ | Radio transmitting @ 4 dBm output power, 1 Mbps Bluetooth low energy mode, Clock = HFXO, Regulator = DCDC | | 8 | | mA |
| $I_{RADIO\_TX1}$ | Radio transmitting @ 0 dBm output power, 1 Mbps Bluetooth low energy mode, Clock = HFXO, Regulator = DCDC | | 5.8 | | mA |
| $I_{RADIO\_TX2}$ | Radio transmitting @ -40 dBm output power, 1 Mbps Bluetooth low energy mode, Clock = HFXO, Regulator = DCDC | | 3.4 | | mA |
| $I_{RADIO\_RX0}$ | Radio receiving @ 1 Mbps Bluetooth low energy mode, Clock = HFXO, Regulator = DCDC | | 6.1 | | mA |
| $I_{RADIO\_TX3}$ | Radio transmitting @ 0 dBm output power, 1 Mbps Bluetooth low energy mode, Clock = HFXO | | 10.5 | | mA |
| $I_{RADIO\_TX4}$ | Radio transmitting @ -40 dBm output power, 1 Mbps Bluetooth low energy mode, Clock = HFXO | | 5.1 | | mA |
| $I_{RADIO\_RX1}$ | Radio receiving @ 1 Mbps Bluetooth low energy mode, Clock = HFXO | | 10.8 | | mA |

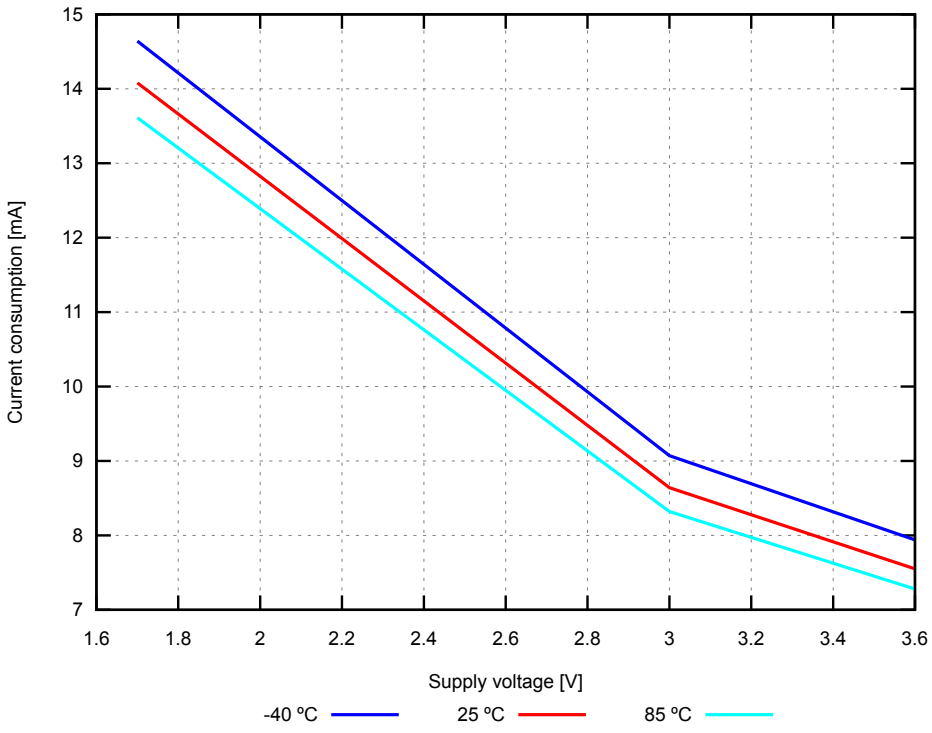



*Figure 9: Radio transmitting @ 4 dBm output power, 1 Mbps Bluetooth low energy mode, Clock = HFXO, Regulator = DCDC (typical values)*

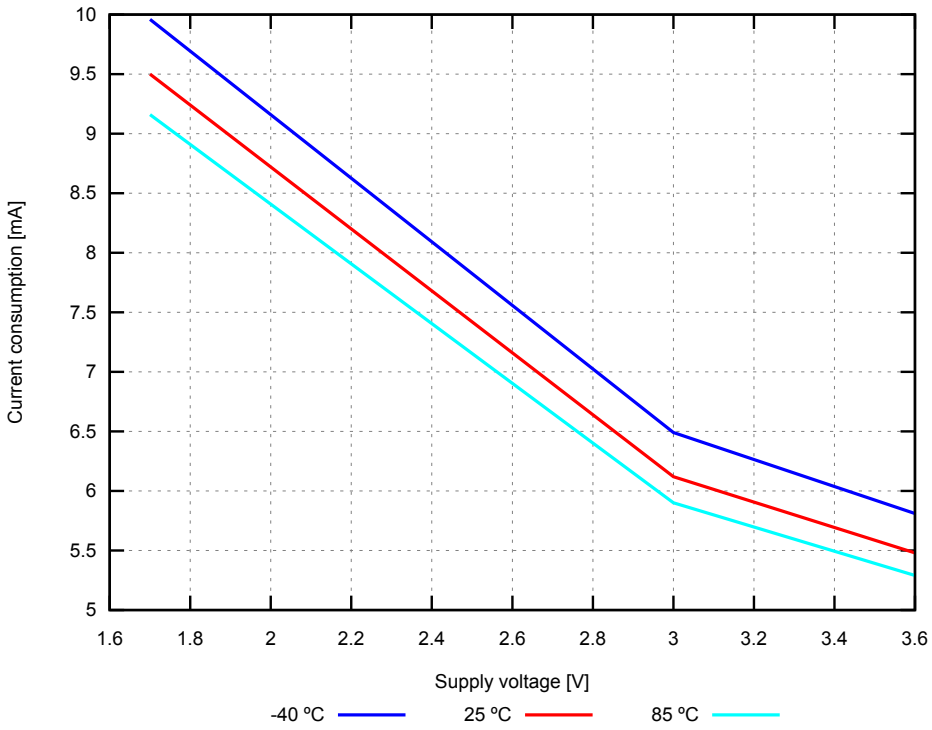



*Figure 10: Radio transmitting @ 0 dBm output power, 1 Mbps Bluetooth low energy mode, Clock = HFXO, Regulator = DCDC (typical values)*

### 5.2.1.3 Sleep

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $I_{ON\_RAMOFF\_EVENT}$ | System ON, No RAM retention, Wake on any event | | 0.6 | | µA |
| $I_{ON\_RAMON\_EVENT}$ | System ON, Full 24 kB RAM retention, Wake on any event | | 0.8 | | µA |
| $I_{ON\_RAMON\_POF}$ | System ON, Full 24 kB RAM retention, Wake on any event, Power fail comparator enabled | | 0.8 | | µA |
| $I_{ON\_RAMON\_GPIOTE}$ | System ON, Full 24 kB RAM retention, Wake on GPIOTE input (Event mode) | | 3.3 | | µA |
| $I_{ON\_RAMON\_GPIOTEPORT}$ | System ON, Full 24 kB RAM retention, Wake on GPIOTE PORT event | | 0.8 | | µA |
| $I_{ON\_RAMON\_RTC}$ | System ON, Full 24 kB RAM retention, Wake on RTC (running from LFRC clock) | | 1.5 | | µA |
| $I_{ON\_RAMOFF\_RTC}$ | System ON, No RAM retention, Wake on RTC (running from LFRC clock) | | 1.4 | | µA |
| $I_{ON\_RAMON\_RTC\_LFXO}$ | System ON, Full 24 kB RAM retention, Wake on RTC (running from LFXO clock) | | 1.1 | | µA |
| $I_{ON\_RAMOFF\_RTC\_LFXO}$ | System ON, No RAM retention, Wake on RTC (running from LFXO clock) | | 1.0 | | µA |
| $I_{OFF\_RAMOFF\_RESET}$ | System OFF, No RAM retention, Wake on reset | | 0.3 | | µA |
| $I_{OFF\_RAMON\_RESET}$ | System OFF, Full 24 kB RAM retention, Wake on reset | | 0.5 | | µA |

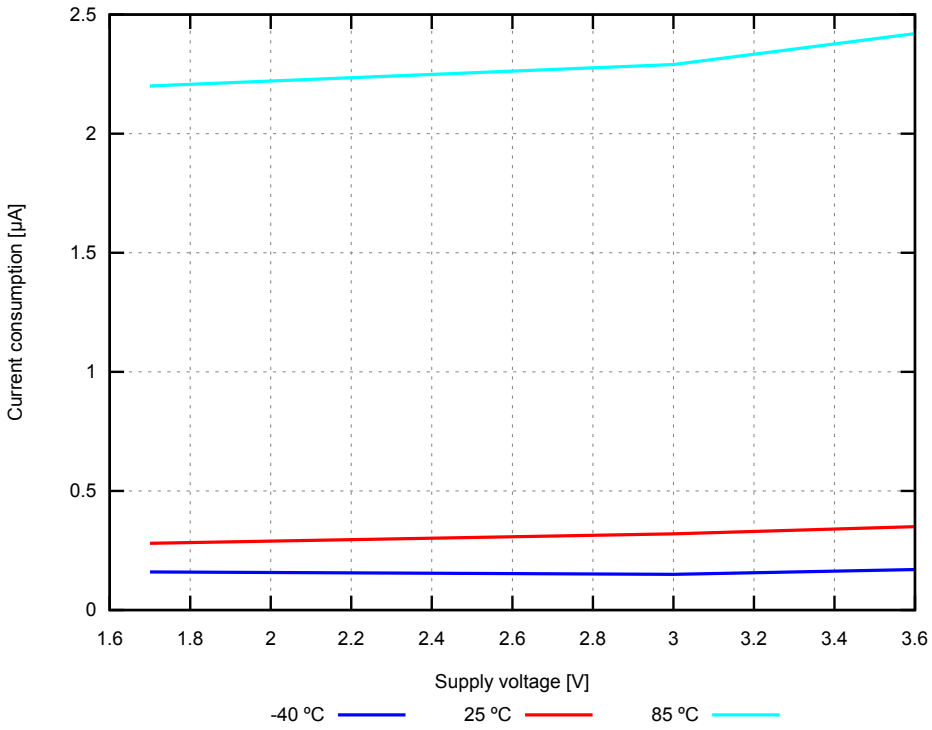



*Figure 11: System OFF, No RAM retention, Wake on reset (typical values)*

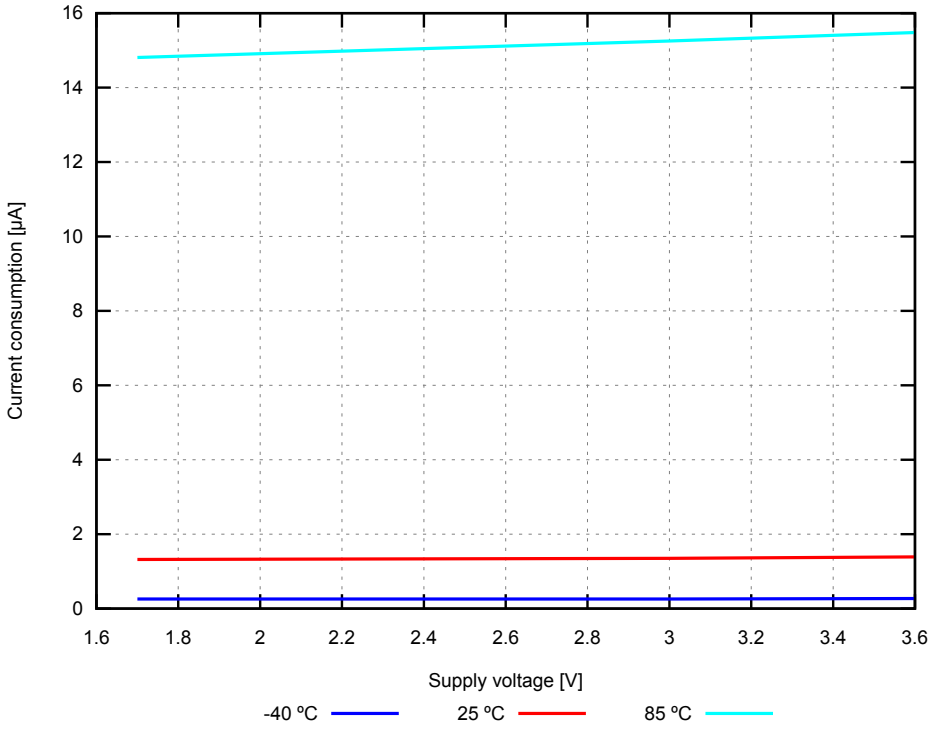



*Figure 12: System ON, Full 24 kB RAM retention, Wake on any event (typical values)*

### 5.2.1.4 Compounded

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $I_{S0}$ | CPU running CoreMark from flash, Radio transmitting @ 0 dBm output power, 1 Mbps Bluetooth low energy mode, Clock = HFXO, Regulator = DCDC | | 7.4 | | mA |
| $I_{S1}$ | CPU running CoreMark from flash, Radio receiving @ 1 Mbps Bluetooth low energy mode, Clock = HFXO, Regulator = DCDC | | 7.6 | | mA |
| $I_{S2}$ | CPU running CoreMark from flash, Radio transmitting @ 0 dBm output power, 1 Mbps Bluetooth low energy mode, Clock = HFXO | | 13.8 | | mA |
| $I_{S3}$ | CPU running CoreMark from flash, Radio receiving @ 1 Mbps Bluetooth low energy mode, Clock = HFXO | | 14.2 | | mA |

### 5.2.1.5 TIMER running

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $I_{TIMER0}$ | One TIMER instance running @ 1 MHz, Clock = HFINT | | 432 | | µA |
| $I_{TIMER1}$ | Two TIMER instances running @ 1 MHz, Clock = HFINT | | 432 | | µA |
| $I_{TIMER2}$ | One TIMER instance running @ 1 MHz, Clock = HFXO | | 730 | | µA |
| $I_{TIMER3}$ | One TIMER instance running @ 16 MHz, Clock = HFINT | | 495 | | µA |
| $I_{TIMER4}$ | One TIMER instance running @ 16 MHz, Clock = HFXO | | 792 | | µA |

### 5.2.1.6 RNG active

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $I_{RNG0}$ | RNG running | | 539 | | µA |

### 5.2.1.7 TEMP active

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $I_{TEMP0}$ | TEMP started | | 998 | | µA |

### 5.2.1.8 SAADC active

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $I_{SAADC,RUN}$ | SAADC sampling @ 16 ksps, Acquisition time = 20 µs, Clock = HFXO, Regulator = DCDC | | 1.1 | | mA |

### 5.2.1.9 COMP active

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $I_{COMP,LP}$ | COMP enabled, low power mode | | 17.2 | | µA |
| $I_{COMP,NORM}$ | COMP enabled, normal mode | | 21 | | µA |
| $I_{COMP,HS}$ | COMP enabled, high-speed mode | | 28.7 | | µA |

#### 5.2.1.10 WDT active

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $I_{WDT,STARTED}$ | WDT started | | 1.3 | | µA |

## 5.3 POWER — Power supply

This device has the following power supply features:

- On-chip LDO and DC/DC regulators
- Global System ON/OFF modes with individual RAM section power control
- Analog or digital pin wakeup from System OFF
- Supervisor HW to manage power on reset, brownout, and power fail
- Auto-controlled refresh modes for LDO and DC/DC regulators to maximize efficiency
- Automatic switching between LDO and DC/DC regulator based on load to maximize efficiency

**Note:** Two additional external passive components are required to use the DC/DC regulator.

### 5.3.1 Regulators

The following internal power regulator alternatives are supported:

- Internal LDO regulator
- Internal DC/DC regulator

The LDO is the default regulator.

The DC/DC regulator can be used as an alternative to the LDO regulator and is enabled through the DCDCEN on page 59 register. Using the DC/DC regulator will reduce current consumption compared to when using the LDO regulator, but the DC/DC regulator requires an external LC filter to be connected, as shown in DC/DC regulator setup on page 49.

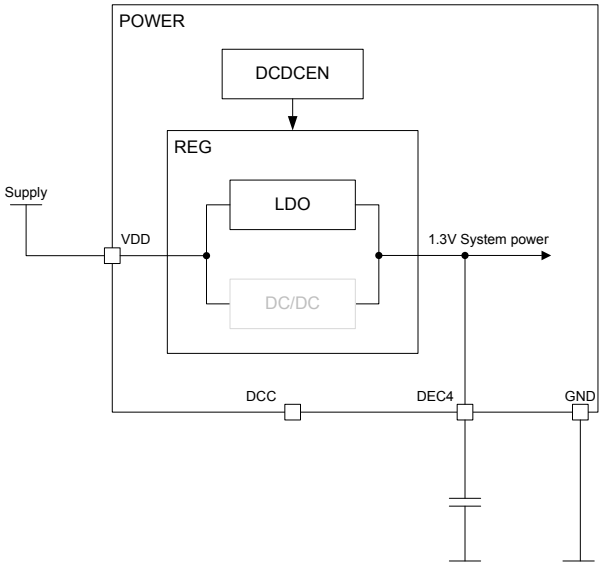



*Figure 13: LDO regulator setup*

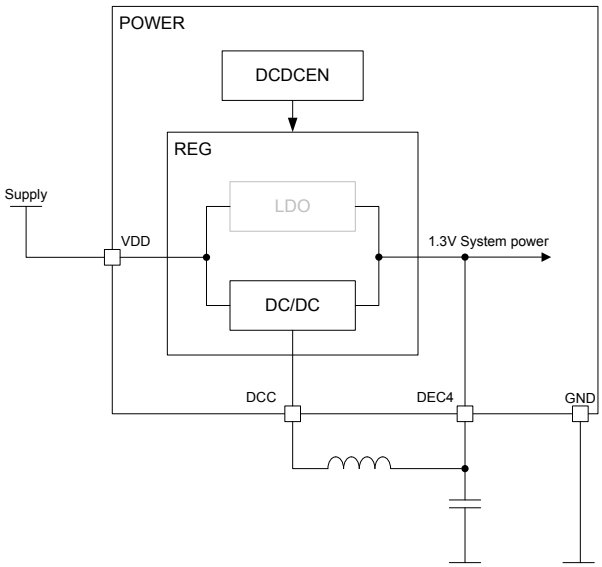



*Figure 14: DC/DC regulator setup*

## 5.3.2 System OFF mode

System OFF is the deepest power saving mode the system can enter. In this mode, the system's core functionality is powered down and all ongoing tasks are terminated.

The device can be put into System OFF mode using the register SYSTEMOFF on page 57. When in System OFF mode, the device can be woken up through one of the following signals:

1. The DETECT signal, optionally generated by the GPIO peripheral
2. A reset

When the system wakes up from System OFF mode, it gets reset. For more details, see Reset behavior on page 53.

One or more RAM sections can be retained in System OFF mode, depending on the settings in the RAM[n].POWER registers.

RAM[n].POWER are retained registers, see Reset behavior. Note that these registers are usually overwritten by the startup code provided with the nRF application examples.

Before entering System OFF mode, the user must make sure that all on-going EasyDMA transactions have been completed. This is usually accomplished by making sure that the EasyDMA enabled peripheral is not active when entering System OFF.

### 5.3.2.1 Emulated System OFF mode

If the device is in debug interface mode, System OFF will be emulated to secure that all required resources needed for debugging are available during System OFF.

See Debug on page 38 for more information. Required resources needed for debugging include the following key components: Debug on page 38, CLOCK — Clock control on page 61, POWER — Power supply on page 48, NVMC — Non-volatile memory controller on page 19, CPU, Flash, and RAM. Since the CPU is kept on in an emulated System OFF mode, it is recommended to add an infinite loop directly after entering System OFF, to prevent the CPU from executing code that normally should not be executed.

### 5.3.3 System ON mode

System ON is the default state after power-on reset. In System ON, all functional blocks such as the CPU or peripherals can be in IDLE or RUN mode, depending on the configuration set by the software and the state of the application executing.

Register RESETREAS on page 57 provides information about the source causing the wakeup or reset.

The system can switch the appropriate internal power sources on and off, depending on how much power is needed at any given time. The power requirement of a peripheral is directly related to its activity level, and the activity level of a peripheral is usually raised and lowered when specific tasks are triggered or events are generated.

#### 5.3.3.1 Sub power modes

In System ON mode, when both the CPU and all the peripherals are in IDLE mode, the system can reside in one of the two sub power modes.

The sub power modes are:

- Constant Latency
- Low-power

In Constant Latency mode, the CPU wakeup latency and the PPI task response are constant and kept at a minimum. This is secured by forcing a set of basic resources to be turned on while in sleep. Having a constant and predictable latency is at the cost of having increased power consumption. The Constant Latency mode is selected by triggering the CONSTLAT task.

In Low-power mode, the automatic power management system described in System ON mode on page 50 ensures that the most efficient supply option is chosen to save most power. Having the lowest power possible is at the cost of having a varying CPU wakeup latency and PPI task response. The Low-power mode is selected by triggering the LOWPWR task.

When the system enters System ON mode, it is by default in Low-power sub power mode.

### 5.3.4 Power supply supervisor

The power supply supervisor initializes the system at power-on and provides an early warning of impending power failure.

In addition, the power supply supervisor puts the system in a reset state if the supply voltage is too low for safe operation (brownout). The power supply supervisor is illustrated in Power supply supervisor on page 51.

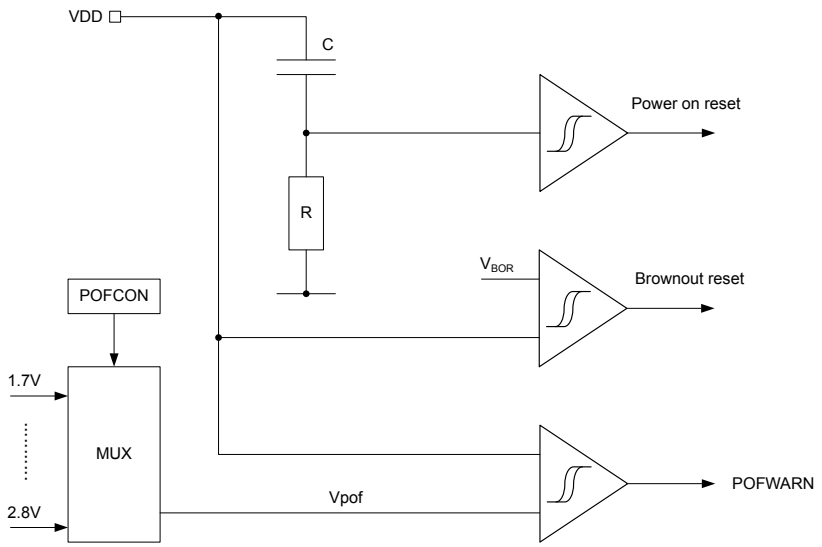



*Figure 15: Power supply supervisor*

### 5.3.4.1 Power-fail comparator

The power-fail comparator (POF) can provide the CPU with an early warning of impending power failure. It will not reset the system, but give the CPU time to prepare for an orderly power-down.

The comparator features a hysteresis of $V_{HYST}$, as illustrated in Power-fail comparator (BOR = Brownout reset) on page 51. The threshold $V_{POF}$ is set in register POFCON on page 58. If the POF is enabled and the supply voltage falls below $V_{POF}$, the POFWARN event will be generated. This event will also be generated if the supply voltage is already below $V_{POF}$ at the time the POF is enabled, or if $V_{POF}$ is re-configured to a level above the supply voltage.

If power-fail warning is enabled and the supply voltage is below $V_{POF}$ the power-fail comparator will prevent the NVMC from performing write operations to the NVM. See NVMC — Non-volatile memory controller on page 19 for more information about the NVMC.

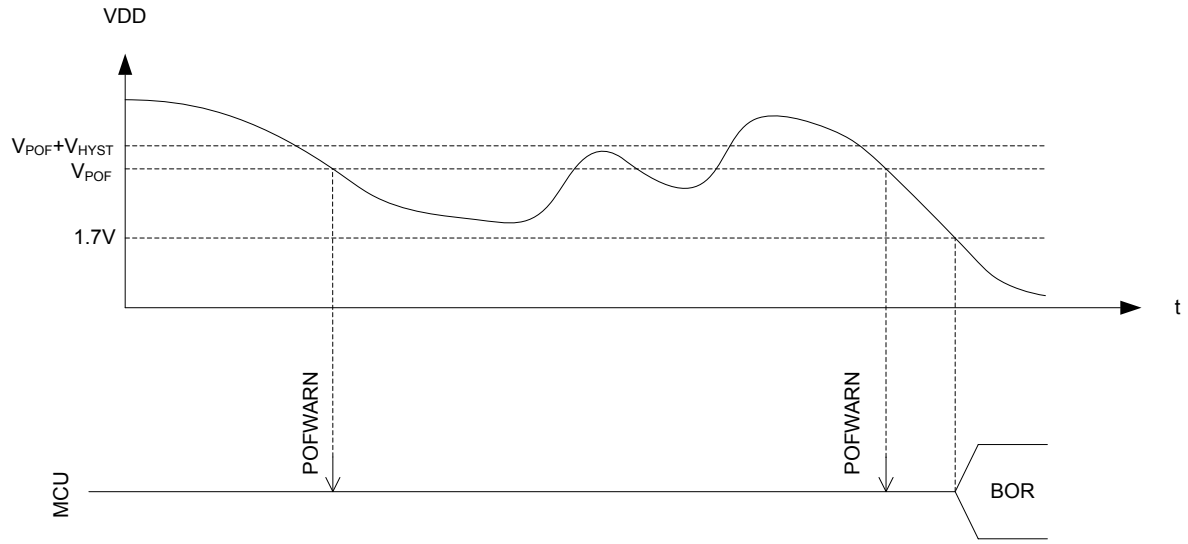



*Figure 16: Power-fail comparator (BOR = Brownout reset)*

To save power, the power-fail comparator is not active in System OFF or in System ON when HFCLK is not running.

## 5.3.5 RAM power control

The RAM power control registers are used for configuring the following:

- The RAM sections to be retained during System OFF
- The RAM sections to be retained and accessible during System ON

In System OFF, retention of a RAM section is configured in the RETENTION field of the corresponding RAM[n] register.

In System ON, retention and accessibility for a RAM section is configured in the RETENTION and POWER fields of the corresponding RAM[n] register.

The following table summarizes the behavior of these registers.

| Configuration | | | RAM section status | |
|---|---|---|---|---|
| System on/off | RAM[n].POWER.POWER | RAM[n].POWER.RETENTION | Accessible | Retained |
| Off | x | Off | No | No |
| Off | x | On | No | Yes |
| On | Off | Off | No | No |
| On | Off[1] | On | No | Yes |
| On | On | x | Yes | Yes |

*Table 14: RAM section configuration. x = don't care.*

The advantage of not retaining RAM contents is that the overall current consumption is reduced.

See chapter Memory on page 16 for more information on RAM sections.

## 5.3.6 Reset

There are multiple sources that may trigger a reset.

After a reset has occurred, register RESETREAS can be read to determine which source generated the reset.

### 5.3.6.1 Power-on reset

The power-on reset generator initializes the system at power-on.

The system is held in reset state until the supply has reached the minimum operating voltage and the internal voltage regulators have started.

A step increase in supply voltage of 300 mV or more, with rise time of 300 ms or less, within the valid supply range, may result in a system reset.

### 5.3.6.2 Pin reset

A pin reset is generated when the physical reset pin on the device is asserted.

Pin reset is configured via the PSELRESET[n] registers.

**Note:** Pin reset is not available on all pins.

### 5.3.6.3 Wakeup from System OFF mode reset

The device is reset when it wakes up from System OFF mode.

[1] Not useful setting. RAM section power off gives negligible reduction in current consumption when retention is on.

The debug access port (DAP) is not reset following a wake up from System OFF mode if the device is in Debug Interface mode. See chapter Debug on page 38 for more information.

#### 5.3.6.4 Soft reset

A soft reset is generated when the SYSRESETREQ bit of the Application Interrupt and Reset Control Register (AIRCR register) in the ARM® core is set.

Refer to ARM documentation for more details.

A soft reset can also be generated via the RESET on page 39 register in the CTRL-AP.

#### 5.3.6.5 Watchdog reset

A Watchdog reset is generated when the watchdog times out.

See chapter WDT — Watchdog timer on page 410 for more information.

#### 5.3.6.6 Brown-out reset

The brown-out reset generator puts the system in reset state if the supply voltage drops below the brownout reset (BOR) threshold.

Refer to section Power fail comparator on page 61 for more information.

### 5.3.7 Retained registers

A retained register is a register that will retain its value in System OFF mode and through a reset, depending on reset source. See individual peripheral chapters for information of which registers are retained for the various peripherals.

### 5.3.8 Reset behavior

| Reset source | Reset target | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | CPU | Peripherals | GPIO | Debug[a] | SWJ-DP | RAM | WDT | Retained registers | RESETREAS |
| CPU lockup [5] | x | x | x | | | | | | |
| Soft reset | x | x | x | | | | | | |
| Wakeup from System OFF mode reset | x | x | | x [6] | | x [7] | | | |
| Watchdog reset [8] | x | x | x | x | | x | x | x | |
| Pin reset | x | x | x | x | | x | x | x | |
| Brownout reset | x | x | x | x | x | x | x | x | x |
| Power on reset | x | x | x | x | x | x | x | x | x |

**Note:** The RAM is never reset, but depending on reset source, RAM content may be corrupted.

[a] All debug components excluding SWJ-DP. See Debug on page 38 chapter for more information about the different debug components in the system.

[5] Reset from CPU lockup is disabled if the device is in debug interface mode. CPU lockup is not possible in System OFF.

[6] The Debug components will not be reset if the device is in debug interface mode.

[7] RAM is not reset on wakeup from OFF mode, but depending on settings in the RAM register parts, or the whole RAM, may not be retained after the device has entered System OFF mode.

[8] Watchdog reset is not available in System OFF.

## 5.3.9 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40000000 | POWER | POWER | Power control | For 24 kB RAM variant, only RAM[0].x to RAM[2].x registers are in use. |

*Table 15: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_CONSTLAT | 0x078 | Enable Constant Latency mode |
| TASKS_LOWPWR | 0x07C | Enable Low-power mode (variable latency) |
| EVENTS_POFWARN | 0x108 | Power failure warning |
| EVENTS_SLEEPENTER | 0x114 | CPU entered WFI/WFE sleep |
| EVENTS_SLEEPEXIT | 0x118 | CPU exited WFI/WFE sleep |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| RESETREAS | 0x400 | Reset reason |
| SYSTEMOFF | 0x500 | System OFF register |
| POFCON | 0x510 | Power failure comparator configuration |
| GPREGRET | 0x51C | General purpose retention register |
| GPREGRET2 | 0x520 | General purpose retention register |
| DCDCEN | 0x578 | DC/DC enable register |
| RAM[0].POWER | 0x900 | RAM0 power control register. The RAM size will vary depending on product variant, and the RAM0 register will only be present if the corresponding RAM AHB slave is present on the device. |
| RAM[0].POWERSET | 0x904 | RAM0 power control set register |
| RAM[0].POWERCLR | 0x908 | RAM0 power control clear register |
| RAM[1].POWER | 0x910 | RAM1 power control register. The RAM size will vary depending on product variant, and the RAM1 register will only be present if the corresponding RAM AHB slave is present on the device. |
| RAM[1].POWERSET | 0x914 | RAM1 power control set register |
| RAM[1].POWERCLR | 0x918 | RAM1 power control clear register |
| RAM[2].POWER | 0x920 | RAM2 power control register. The RAM size will vary depending on product variant, and the RAM2 register will only be present if the corresponding RAM AHB slave is present on the device. |
| RAM[2].POWERSET | 0x924 | RAM2 power control set register |
| RAM[2].POWERCLR | 0x928 | RAM2 power control clear register |
| RAM[3].POWER | 0x930 | RAM3 power control register. The RAM size will vary depending on product variant, and the RAM3 register will only be present if the corresponding RAM AHB slave is present on the device. |
| RAM[3].POWERSET | 0x934 | RAM3 power control set register |
| RAM[3].POWERCLR | 0x938 | RAM3 power control clear register |
| RAM[4].POWER | 0x940 | RAM4 power control register. The RAM size will vary depending on product variant, and the RAM4 register will only be present if the corresponding RAM AHB slave is present on the device. |
| RAM[4].POWERSET | 0x944 | RAM4 power control set register |
| RAM[4].POWERCLR | 0x948 | RAM4 power control clear register |
| RAM[5].POWER | 0x950 | RAM5 power control register. The RAM size will vary depending on product variant, and the RAM5 register will only be present if the corresponding RAM AHB slave is present on the device. |
| RAM[5].POWERSET | 0x954 | RAM5 power control set register |
| RAM[5].POWERCLR | 0x958 | RAM5 power control clear register |

| Register | Offset | Description |
|---|---|---|
| RAM[6].POWER | 0x960 | RAM6 power control register. The RAM size will vary depending on product variant, and the RAM6 register will only be present if the corresponding RAM AHB slave is present on the device. |
| RAM[6].POWERSET | 0x964 | RAM6 power control set register |
| RAM[6].POWERCLR | 0x968 | RAM6 power control clear register |
| RAM[7].POWER | 0x970 | RAM7 power control register. The RAM size will vary depending on product variant, and the RAM7 register will only be present if the corresponding RAM AHB slave is present on the device. |
| RAM[7].POWERSET | 0x974 | RAM7 power control set register |
| RAM[7].POWERCLR | 0x978 | RAM7 power control clear register |

*Table 16: Register overview*

### 5.3.9.1 TASKS_CONSTLAT

Address offset: 0x078

Enable Constant Latency mode

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_CONSTLAT | | | Enable Constant Latency mode |
| | | | Trigger | 1 | Trigger task |

### 5.3.9.2 TASKS_LOWPWR

Address offset: 0x07C

Enable Low-power mode (variable latency)

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_LOWPWR | | | Enable Low-power mode (variable latency) |
| | | | Trigger | 1 | Trigger task |

### 5.3.9.3 EVENTS_POFWARN

Address offset: 0x108

Power failure warning

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_POFWARN | | | Power failure warning |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 5.3.9.4 EVENTS_SLEEPENTER

Address offset: 0x114

CPU entered WFI/WFE sleep

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_SLEEPENTER | | | CPU entered WFI/WFE sleep |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 5.3.9.5 EVENTS_SLEEPEXIT

Address offset: 0x118

CPU exited WFI/WFE sleep

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_SLEEPEXIT | | | CPU exited WFI/WFE sleep |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 5.3.9.6 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | POFWARN | | | Write '1' to enable interrupt for event POFWARN |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | SLEEPENTER | | | Write '1' to enable interrupt for event SLEEPENTER |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | SLEEPEXIT | | | Write '1' to enable interrupt for event SLEEPEXIT |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

### 5.3.9.7 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | POFWARN | | | Write '1' to disable interrupt for event POFWARN |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | SLEEPENTER | | | Write '1' to disable interrupt for event SLEEPENTER |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | SLEEPEXIT | | | Write '1' to disable interrupt for event SLEEPEXIT |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 5.3.9.8 RESETREAS

Address offset: 0x400

Reset reason

Unless cleared, the RESETREAS register will be cumulative. A field is cleared by writing '1' to it. If none of the reset sources are flagged, this indicates that the chip was reset from the on-chip reset generator, which will indicate a power-on-reset or a brownout reset.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | RESETPIN | | | Reset from pin-reset detected |
| | | | NotDetected | 0 | Not detected |
| | | | Detected | 1 | Detected |
| B | RW | DOG | | | Reset from watchdog detected |
| | | | NotDetected | 0 | Not detected |
| | | | Detected | 1 | Detected |
| C | RW | SREQ | | | Reset from soft reset detected |
| | | | NotDetected | 0 | Not detected |
| | | | Detected | 1 | Detected |
| D | RW | LOCKUP | | | Reset from CPU lock-up detected |
| | | | NotDetected | 0 | Not detected |
| | | | Detected | 1 | Detected |
| E | RW | OFF | | | Reset due to wake up from System OFF mode when wakeup is triggered from DETECT signal from GPIO |
| | | | NotDetected | 0 | Not detected |
| | | | Detected | 1 | Detected |
| F | RW | DIF | | | Reset due to wake up from System OFF mode when wakeup is triggered from entering into debug interface mode |
| | | | NotDetected | 0 | Not detected |
| | | | Detected | 1 | Detected |

## 5.3.9.9 SYSTEMOFF

Address offset: 0x500

System OFF register

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | SYSTEMOFF | | | Enable System OFF mode |
| | | | Enter | 1 | Enable System OFF mode |

## 5.3.9.10 POFCON

Address offset: 0x510

Power failure comparator configuration

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | B B B B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | POF | | | Enable or disable power failure comparator |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| B | RW | THRESHOLD | | | Power failure comparator threshold setting |
| | | | V17 | 4 | Set threshold to 1.7 V |
| | | | V18 | 5 | Set threshold to 1.8 V |
| | | | V19 | 6 | Set threshold to 1.9 V |
| | | | V20 | 7 | Set threshold to 2.0 V |
| | | | V21 | 8 | Set threshold to 2.1 V |
| | | | V22 | 9 | Set threshold to 2.2 V |
| | | | V23 | 10 | Set threshold to 2.3 V |
| | | | V24 | 11 | Set threshold to 2.4 V |
| | | | V25 | 12 | Set threshold to 2.5 V |
| | | | V26 | 13 | Set threshold to 2.6 V |
| | | | V27 | 14 | Set threshold to 2.7 V |
| | | | V28 | 15 | Set threshold to 2.8 V |

## 5.3.9.11 GPREGRET

Address offset: 0x51C

General purpose retention register

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | GPREGRET | | | General purpose retention register<br><br>This register is a retained register |

## 5.3.9.12 GPREGRET2

Address offset: 0x520

General purpose retention register

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | GPREGRET | | | General purpose retention register<br>This register is a retained register |

## 5.3.9.13 DCDCEN

Address offset: 0x578

DC/DC enable register

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | DCDCEN | | | Enable or disable DC/DC converter |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |

## 5.3.9.14 RAM[n].POWER (n=0..7)

Address offset: 0x900 + (n × 0x10)

RAMn power control register. The RAM size will vary depending on product variant, and the RAMn register will only be present if the corresponding RAM AHB slave is present on the device.

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | D C B A |
| **Reset 0x0000FFFF** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A-B | RW | S[i]POWER (i=0..1) | | | Keep RAM section Si ON or OFF in System ON mode.<br>RAM sections are always retained when ON, but can also be retained when OFF dependent on the settings in SiRETENTION. All RAM sections will be OFF in System OFF mode. |
| | | | Off | 0 | Off |
| | | | On | 1 | On |
| C-D | RW | S[i]RETENTION (i=0..1) | | | Keep retention on RAM section Si when RAM section is in OFF |
| | | | Off | 0 | Off |
| | | | On | 1 | On |

## 5.3.9.15 RAM[n].POWERSET (n=0..7)

Address offset: 0x904 + (n × 0x10)

RAMn power control set register

When read, this register will return the value of the POWER register.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | D C B A |
| **Reset 0x0000FFFF** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A-B | W | S[i]POWER (i=0..1) | | | Keep RAM section Si of RAMn on or off in System ON mode |
| | | | On | 1 | On |
| C-D | W | S[i]RETENTION (i=0..1) | | | Keep retention on RAM section Si when RAM section is switched off |
| | | | On | 1 | On |

### 5.3.9.16 RAM[n].POWERCLR (n=0..7)

Address offset: 0x908 + (n × 0x10)

RAMn power control clear register

When read, this register will return the value of the POWER register.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | D C B A |
| **Reset 0x0000FFFF** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A-B | W | S[i]POWER (i=0..1) | | | Keep RAM section Si of RAMn on or off in System ON mode |
| | | | Off | 1 | Off |
| C-D | W | S[i]RETENTION (i=0..1) | | | Keep retention on RAM section Si when RAM section is switched off |
| | | | Off | 1 | Off |

## 5.3.10 Electrical specification

### 5.3.10.1 Device startup times

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{POR}$ | Time in Power on Reset after VDD reaches 1.7 V for all supply voltages and temperatures. Dependent on supply rise time. [9] | | | | |
| $t_{POR,10us}$ | VDD rise time 10us | | 1 | | ms |
| $t_{POR,10ms}$ | VDD rise time 10ms | | 9 | | ms |
| $t_{POR,60ms}$ | VDD rise time 60ms | | 23 | | ms |
| $t_{PINR}$ | If a GPIO pin is configured as reset, the maximum time taken to pull up the pin and release reset after power on reset. Dependent on the pin capacitive load (C)[10]: t=5RC, R = 13kOhm | | | | |
| $t_{PINR,500nF}$ | C = 500nF | | | 32.5 | ms |
| $t_{PINR,10uF}$ | C = 10uF | | | 650 | ms |
| $t_{R2ON}$ | Time from reset to ON (CPU execute) | | | | |
| $t_{R2ON,NOTCONF}$ | If reset pin not configured | tPOR | | | ms |
| $t_{R2ON,CONF}$ | If reset pin configured | tPOR + tPINR | | | ms |

[9] A step increase in supply voltage of 300 mV or more, with rise time of 300 ms or less, within the valid supply range, may result in a system reset.

[10] To decrease maximum time a device could hold in reset, a strong external pullup resistor can be used.

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{OFF2ON}$ | Time from OFF to CPU execute | | 16.5 | | µs |
| $t_{IDLE2CPU}$ | Time from IDLE to CPU execute | | 3.0 | | µs |
| $t_{EVTSET,CL1}$ | Time from HW event to PPI event in Constant Latency System ON mode | | 0.0625 | | µs |
| $t_{EVTSET,CL0}$ | Time from HW event to PPI event in Low Power System ON mode | | 0.0625 | | µs |

### 5.3.10.2 Power fail comparator

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $V_{POF}$ | Nominal power level warning thresholds (falling supply voltage). Levels are configurable between Min. and Max. in 100mV increments. | 1.7 | | 2.8 | V |
| $V_{POFTOL}$ | Threshold voltage tolerance | | ±1 | ±5 | % |
| $V_{POFHYST}$ | Threshold voltage hysteresis | | 50 | | mV |
| $V_{BOR,OFF}$ | Brown out reset voltage range SYSTEM OFF mode | 1.2 | | 1.7 | V |
| $V_{BOR,ON}$ | Brown out reset voltage range SYSTEM ON mode | 1.48 | | 1.7 | V |

## 5.4 CLOCK — Clock control

The clock control system can source the system clocks from a range of internal or external high and low frequency oscillators and distribute them to modules based upon a module's individual requirements. Clock distribution is automated and grouped independently by module to limit current consumption in unused branches of the clock tree.

Listed here are the main features for CLOCK:

- 64 MHz on-chip oscillator
- 64 MHz crystal oscillator, using external 32 MHz crystal
- 32.768 kHz +/-500 ppm RC oscillator
- 32.768 kHz crystal oscillator, using external 32.768 kHz crystal
- 32.768 kHz oscillator synthesized from 64 MHz oscillator
- Firmware (FW) override control of oscillator activity for low latency start up
- Automatic oscillator and clock control, and distribution for ultra-low power

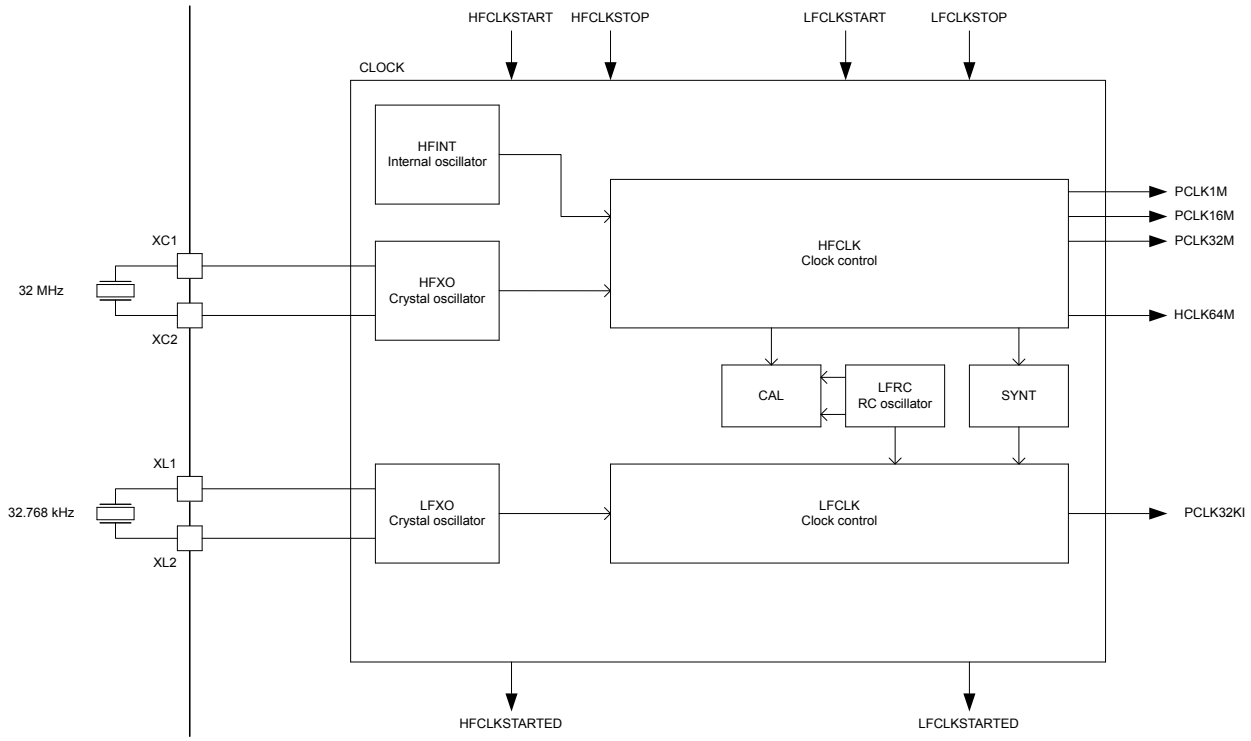



*Figure 17: Clock control*

## 5.4.1 HFCLK clock controller

The HFCLK clock controller provides the following clocks to the system.

- HCLK64M: 64 MHz CPU clock
- PCLK1M: 1 MHz peripheral clock
- PCLK16M: 16 MHz peripheral clock
- PCLK32M: 32 MHz peripheral clock

The HFCLK controller supports the following high frequency clock (HFCLK) sources:

- 64 MHz internal oscillator (HFINT)
- 64 MHz crystal oscillator (HFXO)

For illustration, see Clock control on page 62.

When the system requests one or more clocks from the HFCLK controller, the HFCLK controller will automatically provide them. If the system does not request any clocks provided by the HFCLK controller, the controller will enter a power saving mode.

These clocks are only available when the system is in ON mode. When the system enters ON mode, the internal oscillator (HFINT) clock source will automatically start to be able to provide the required HFCLK clock(s) for the system.

The HFINT will be used when HFCLK is requested and HFXO has not been started. The HFXO is started by triggering the HFCLKSTART task and stopped using the HFCLKSTOP task. A HFCLKSTARTED event will be generated when the HFXO has started and its frequency is stable.

The HFXO must be running to use the RADIO or the calibration mechanism associated with the 32.768 kHz RC oscillator.

### 5.4.1.1 64 MHz crystal oscillator (HFXO)

The 64 MHz crystal oscillator (HFXO) is controlled by a 32 MHz external crystal

The crystal oscillator is designed for use with an AT-cut quartz crystal in parallel resonant mode. To achieve correct oscillation frequency, the load capacitance must match the specification in the crystal data sheet.

Circuit diagram of the 64 MHz crystal oscillator on page 63 shows how the 32 MHz crystal is connected to the 64 MHz crystal oscillator.

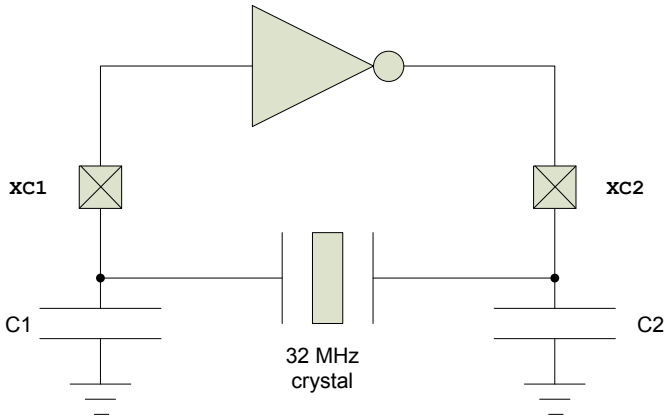



*Figure 18: Circuit diagram of the 64 MHz crystal oscillator*

The load capacitance (CL) is the total capacitance seen by the crystal across its terminals and is given by:

$$CL = \frac{(C1' \cdot C2')}{(C1' + C2')}$$

$$C1' = C1 + C_{pcb1} + C_{pin}$$
$$C2' = C2 + C_{pcb2} + C_{pin}$$

C1 and C2 are ceramic SMD capacitors connected between each crystal terminal and ground. For more information, see Reference circuitry on page 425. $C_{pcb1}$ and $C_{pcb2}$ are stray capacitances on the PCB. $C_{pin}$ is the pin input capacitance on the `XC1` and `XC2` pins. See table 64 MHz crystal oscillator (HFXO) on page 72. The load capacitors C1 and C2 should have the same value.

For reliable operation, the crystal load capacitance, shunt capacitance, equivalent series resistance, and drive level must comply with the specifications in table 64 MHz crystal oscillator (HFXO) on page 72. It is recommended to use a crystal with lower than maximum load capacitance and/or shunt capacitance. A low load capacitance will reduce both start up time and current consumption.

## 5.4.2 LFCLK clock controller

The system supports several low frequency clock sources.

As illustrated in Clock control on page 62, the system supports the following low frequency clock sources:

- 32.768 kHz RC oscillator (LFRC)
- 32.768 kHz crystal oscillator (LFXO)
- 32.768 kHz synthesized from HFCLK (LFSYNT)

The LFCLK clock is started by first selecting the preferred clock source in register LFCLKSRC on page 71 and then triggering the LFCLKSTART task. If the LFXO is selected as the clock source, the LFCLK will initially start running from the 32.768 kHz LFRC while the LFXO is starting up and automatically switch to using the LFXO once this oscillator is running. The LFCLKSTARTED event will be generated when the LFXO has been started.

The LFCLK clock is stopped by triggering the LFCLKSTOP task.

It is not allowed to write to register LFCLKSRC on page 71 when the LFCLK is running.

A LFCLKSTOP task will stop the LFCLK oscillator. However, the LFCLKSTOP task can only be triggered after the STATE field in register LFCLKSTAT on page 71 indicates a 'LFCLK running' state.

The LFCLK clock controller and all of the LFCLK clock sources are always switched off when in OFF mode.

### 5.4.2.1 32.768 kHz RC oscillator (LFRC)

The default source of the low frequency clock (LFCLK) is the 32.768 kHz RC oscillator (LFRC).

The LFRC frequency will be affected by variation in temperature. The LFRC oscillator can be calibrated to improve accuracy by using the HFXO as a reference oscillator during calibration. See Table 32.768 kHz RC oscillator (LFRC) on page 73 for details on the default and calibrated accuracy of the LFRC oscillator.
The LFRC oscillator does not require additional external components.

### 5.4.2.2 Calibrating the 32.768 kHz RC oscillator

After the 32.768 kHz RC oscillator is started and running, it can be calibrated by triggering the CAL task. In this case, the HFCLK will be temporarily switched on and used as a reference.

A DONE event will be generated when calibration has finished. The calibration mechanism will only work as long as HFCLK is generated from the HFCLK crystal oscillator, it is therefore necessary to explicitly start this crystal oscillator before calibration can be started, see HFCLKSTART task.

It is not allowed to stop the LFRC during an ongoing calibration.

### 5.4.2.3 Calibration timer

The calibration timer can be used to time the calibration interval of the 32.768 kHz RC oscillator.

The calibration timer is started by triggering the CTSTART task and stopped by triggering the CTSTOP task. The calibration timer will always start counting down from the value specified in CTIV and generate a CTTO timeout event when it reaches 0. The Calibration timer will stop by itself when it reaches 0.

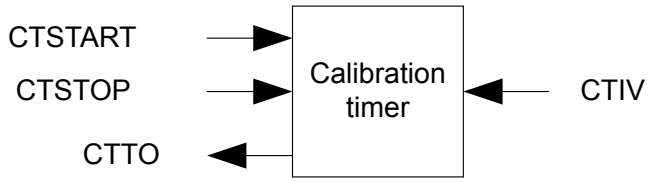



*Figure 19: Calibration timer*

Due to limitations in the calibration timer, only one task related to calibration, that is, CAL, CTSTART and CTSTOP, can be triggered for every period of LFCLK.

### 5.4.2.4 32.768 kHz crystal oscillator (LFXO)

For higher LFCLK accuracy the low frequency crystal oscillator (LFXO) must be used.

The following external clock sources are supported:

- Low swing clock signal applied to the *XL1* pin. The *XL2* pin shall then be grounded.
- Rail-to-rail clock signal applied to the *XL1* pin. The *XL2* pin shall then be grounded or left unconnected.

The LFCLKSRC on page 71 register controls the clock source, and its allowed swing. The truth table for various situations is as follows:

| SRC | EXTERNAL | BYPASS | Comment |
|---|---|---|---|
| 0 | 0 | 0 | Normal operation, RC is source |
| 0 | 0 | 1 | DO NOT USE |
| 0 | 1 | X | DO NOT USE |
| 1 | 0 | 0 | Normal XTAL operation |
| 1 | 1 | 0 | Apply external low swing signal to XL1, ground XL2 |
| 1 | 1 | 1 | Apply external full swing signal to XL1, leave XL2 grounded or unconnected |
| 1 | 0 | 1 | DO NOT USE |
| 2 | 0 | 0 | Normal operation, synth is source |
| 2 | 0 | 1 | DO NOT USE |
| 2 | 1 | X | DO NOT USE |

*Table 17: LFCLKSRC configuration depending on clock source*

To achieve correct oscillation frequency, the load capacitance must match the specification in the crystal data sheet. Circuit diagram of the 32.768 kHz crystal oscillator on page 65 shows the LFXO circuitry.

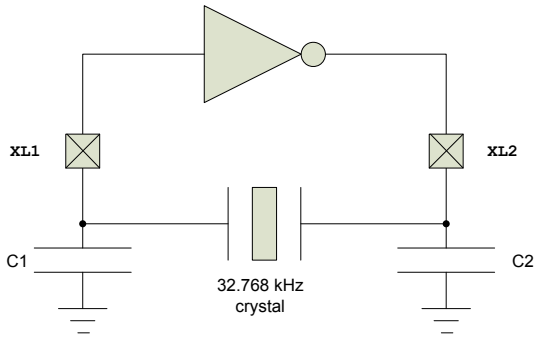



*Figure 20: Circuit diagram of the 32.768 kHz crystal oscillator*

The load capacitance (CL) is the total capacitance seen by the crystal across its terminals and is given by:

$$CL = \frac{(C1' \cdot C2')}{(C1' + C2')}$$

$$C1' = C1 + C_{pcb1} + C_{pin}$$
$$C2' = C2 + C_{pcb2} + C_{pin}$$

C1 and C2 are ceramic SMD capacitors connected between each crystal terminal and ground. $C_{pcb1}$ and $C_{pcb2}$ are stray capacitances on the PCB. $C_{pin}$ is the pin input capacitance on the XC1 and XC2 pins (see 32.768 kHz crystal oscillator (LFXO) on page 73). The load capacitors C1 and C2 should have the same value.

For more information, see Reference circuitry on page 425.

### 5.4.2.5 32.768 kHz synthesized from HFCLK (LFSYNT)

LFCLK can also be synthesized from the HFCLK clock source. The accuracy of LFCLK will then be the accuracy of the HFCLK.

Using the LFSYNT clock avoids the requirement for a 32.768 kHz crystal, but increases average power consumption as the HFCLK will need to be requested in the system.

## 5.4.3 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40000000 | CLOCK | CLOCK | Clock control | |

*Table 18: Instances*

| Register | Offset | Description | |
|---|---|---|---|
| TASKS_HFCLKSTART | 0x000 | Start HFCLK crystal oscillator | |
| TASKS_HFCLKSTOP | 0x004 | Stop HFCLK crystal oscillator | |
| TASKS_LFCLKSTART | 0x008 | Start LFCLK source | |
| TASKS_LFCLKSTOP | 0x00C | Stop LFCLK source | |
| TASKS_CAL | 0x010 | Start calibration of LFRC oscillator | |
| TASKS_CTSTART | 0x014 | Start calibration timer | |
| TASKS_CTSTOP | 0x018 | Stop calibration timer | |
| EVENTS_HFCLKSTARTED | 0x100 | HFCLK oscillator started | |
| EVENTS_LFCLKSTARTED | 0x104 | LFCLK started | |
| EVENTS_DONE | 0x10C | Calibration of LFCLK RC oscillator complete event | |
| EVENTS_CTTO | 0x110 | Calibration timer timeout | |
| INTENSET | 0x304 | Enable interrupt | |
| INTENCLR | 0x308 | Disable interrupt | |
| HFCLKRUN | 0x408 | Status indicating that HFCLKSTART task has been triggered | |
| HFCLKSTAT | 0x40C | HFCLK status | |
| LFCLKRUN | 0x414 | Status indicating that LFCLKSTART task has been triggered | |
| LFCLKSTAT | 0x418 | LFCLK status | |
| LFCLKSRCCOPY | 0x41C | Copy of LFCLKSRC register, set when LFCLKSTART task was triggered | |
| LFCLKSRC | 0x518 | Clock source for the LFCLK | |
| CTIV | 0x538 | Calibration timer interval | Retained |

*Table 19: Register overview*

### 5.4.3.1 TASKS_HFCLKSTART

Address offset: 0x000

Start HFCLK crystal oscillator

| Bit number | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | A |
| **Reset 0x00000000** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_HFCLKSTART | | | Start HFCLK crystal oscillator |
| | | | Trigger | 1 | Trigger task |

### 5.4.3.2 TASKS_HFCLKSTOP

Address offset: 0x004

Stop HFCLK crystal oscillator

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_HFCLKSTOP | | | Stop HFCLK crystal oscillator |
| | | | Trigger | 1 | Trigger task |

### 5.4.3.3 TASKS_LFCLKSTART

Address offset: 0x008

Start LFCLK source

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_LFCLKSTART | | | Start LFCLK source |
| | | | Trigger | 1 | Trigger task |

### 5.4.3.4 TASKS_LFCLKSTOP

Address offset: 0x00C

Stop LFCLK source

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_LFCLKSTOP | | | Stop LFCLK source |
| | | | Trigger | 1 | Trigger task |

### 5.4.3.5 TASKS_CAL

Address offset: 0x010

Start calibration of LFRC oscillator

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_CAL | | | Start calibration of LFRC oscillator |
| | | | Trigger | 1 | Trigger task |

### 5.4.3.6 TASKS_CTSTART

Address offset: 0x014

Start calibration timer

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_CTSTART | | | Start calibration timer |
| | | | Trigger | 1 | Trigger task |

### 5.4.3.7 TASKS_CTSTOP

Address offset: 0x018

Stop calibration timer

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_CTSTOP | | | Stop calibration timer |
| | | | Trigger | 1 | Trigger task |

### 5.4.3.8 EVENTS_HFCLKSTARTED

Address offset: 0x100

HFCLK oscillator started

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_HFCLKSTARTED | | | HFCLK oscillator started |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 5.4.3.9 EVENTS_LFCLKSTARTED

Address offset: 0x104

LFCLK started

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_LFCLKSTARTED | | | LFCLK started |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 5.4.3.10 EVENTS_DONE

Address offset: 0x10C

Calibration of LFCLK RC oscillator complete event

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | EVENTS_DONE | | | Calibration of LFCLK RC oscillator complete event |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 5.4.3.11 EVENTS_CTTO

Address offset: 0x110

Calibration timer timeout

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | EVENTS_CTTO | | | Calibration timer timeout |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 5.4.3.12 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | D C B A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | HFCLKSTARTED | | | Write '1' to enable interrupt for event HFCLKSTARTED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | LFCLKSTARTED | | | Write '1' to enable interrupt for event LFCLKSTARTED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | DONE | | | Write '1' to enable interrupt for event DONE |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | CTTO | | | Write '1' to enable interrupt for event CTTO |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 5.4.3.13 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | HFCLKSTARTED | | | Write '1' to disable interrupt for event HFCLKSTARTED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | LFCLKSTARTED | | | Write '1' to disable interrupt for event LFCLKSTARTED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | DONE | | | Write '1' to disable interrupt for event DONE |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | CTTO | | | Write '1' to disable interrupt for event CTTO |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

### 5.4.3.14 HFCLKRUN

Address offset: 0x408

Status indicating that HFCLKSTART task has been triggered

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | STATUS | | | HFCLKSTART task triggered or not |
| | | | NotTriggered | 0 | Task not triggered |
| | | | Triggered | 1 | Task triggered |

### 5.4.3.15 HFCLKSTAT

Address offset: 0x40C

HFCLK status

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | SRC | | | Source of HFCLK |
| | | | RC | 0 | 64 MHz internal oscillator (HFINT) |
| | | | Xtal | 1 | 64 MHz crystal oscillator (HFXO) |
| B | R | STATE | | | HFCLK state |
| | | | NotRunning | 0 | HFCLK not running |
| | | | Running | 1 | HFCLK running |

### 5.4.3.16 LFCLKRUN

Address offset: 0x414

Status indicating that LFCLKSTART task has been triggered

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | STATUS | | | LFCLKSTART task triggered or not |
| | | | NotTriggered | 0 | Task not triggered |
| | | | Triggered | 1 | Task triggered |

## 5.4.3.17 LFCLKSTAT

Address offset: 0x418

LFCLK status

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | B A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | SRC | | | Source of LFCLK |
| | | | RC | 0 | 32.768 kHz RC oscillator |
| | | | Xtal | 1 | 32.768 kHz crystal oscillator |
| | | | Synth | 2 | 32.768 kHz synthesized from HFCLK |
| B | R | STATE | | | LFCLK state |
| | | | NotRunning | 0 | LFCLK not running |
| | | | Running | 1 | LFCLK running |

## 5.4.3.18 LFCLKSRCCOPY

Address offset: 0x41C

Copy of LFCLKSRC register, set when LFCLKSTART task was triggered

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | SRC | | | Clock source |
| | | | RC | 0 | 32.768 kHz RC oscillator |
| | | | Xtal | 1 | 32.768 kHz crystal oscillator |
| | | | Synth | 2 | 32.768 kHz synthesized from HFCLK |

## 5.4.3.19 LFCLKSRC

Address offset: 0x518

Clock source for the LFCLK

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | C B A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | SRC | | | Clock source |
| | | | RC | 0 | 32.768 kHz RC oscillator |
| | | | Xtal | 1 | 32.768 kHz crystal oscillator |
| | | | Synth | 2 | 32.768 kHz synthesized from HFCLK |
| B | RW | BYPASS | | | Enable or disable bypass of LFCLK crystal oscillator with external clock source |
| | | | Disabled | 0 | Disable (use with Xtal or low-swing external source) |
| | | | Enabled | 1 | Enable (use with rail-to-rail external source) |
| C | RW | EXTERNAL | | | Enable or disable external source for LFCLK |
| | | | Disabled | 0 | Disable external source (use with Xtal) |
| | | | Enabled | 1 | Enable use of external source instead of Xtal (SRC needs to be set to Xtal) |

### 5.4.3.20 CTIV ( Retained )

Address offset: 0x538

This register is a retained register

Calibration timer interval

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | CTIV | | | Calibration timer interval in multiple of 0.25 seconds. Range: 0.25 seconds to 31.75 seconds. |

## 5.4.4 Electrical specification

### 5.4.4.1 64 MHz internal oscillator (HFINT)

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $f_{NOM\_HFINT}$ | Nominal output frequency | | 64 | | MHz |
| $f_{TOL\_HFINT}$ | Frequency tolerance | | <±1.5 | <±8 | % |
| $t_{START\_HFINT}$ | Startup time | | 3 | | us |

### 5.4.4.2 64 MHz crystal oscillator (HFXO)

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $f_{NOM\_HFXO}$ | Nominal output frequency | | 64 | | MHz |
| $f_{XTAL\_HFXO}$ | External crystal frequency | | 32 | | MHz |
| $f_{TOL\_HFXO}$ | Frequency tolerance requirement for 2.4 GHz proprietary radio applications | | | ±60 | ppm |
| $f_{TOL\_HFXO\_BLE}$ | Frequency tolerance requirement, Bluetooth low energy applications | | | ±40 | ppm |
| $C_{L\_HFXO}$ | Load capacitance | | | 12 | pF |
| $C_{0\_HFXO}$ | Shunt capacitance | | | 7 | pF |

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $R_{S\_HFXO\_7PF}$ | Equivalent series resistance C0 = 7 pF | | | 60 | ohm |
| $R_{S\_HFXO\_5PF}$ | Equivalent series resistance C0 = 5 pF | | | 60 | ohm |
| $R_{S\_HFXO\_3PF}$ | Equivalent series resistance C0 = 3 pF | | | 100 | ohm |
| $P_{D\_HFXO}$ | Drive level | | | 100 | uW |
| $C_{PIN\_HFXO}$ | Input capacitance XC1 and XC2 | | 4 | | pF |
| $t_{START\_HFXO}$ | Startup time | | 0.36 | | ms |

### 5.4.4.3 32.768 kHz RC oscillator (LFRC)

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $f_{NOM\_LFRC}$ | Nominal frequency | | 32.768 | | kHz |
| $f_{TOL\_LFRC}$ | Frequency tolerance | | | ±2 | % |
| $f_{TOL\_CAL\_LFRC}$ | Frequency tolerance for LFRC after calibration[11] | | | ±500 | ppm |
| $t_{START\_LFRC}$ | Startup time for 32.768 kHz RC oscillator | | 600 | | us |

### 5.4.4.4 32.768 kHz crystal oscillator (LFXO)

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $f_{NOM\_LFXO}$ | Crystal frequency | | 32.768 | | kHz |
| $f_{TOL\_LFXO\_BLE}$ | Frequency tolerance requirement for BLE stack | | | ±250 | ppm |
| $f_{TOL\_LFXO\_ANT}$ | Frequency tolerance requirement for ANT stack | | | ±50 | ppm |
| $C_{L\_LFXO}$ | Load capacitance | | | 12.5 | pF |
| $C_{0\_LFXO}$ | Shunt capacitance | | | 2 | pF |
| $R_{S\_LFXO}$ | Equivalent series resistance | | | 100 | kohm |
| $P_{D\_LFXO}$ | Drive level | | | 0.5 | uW |
| $C_{pin}$ | Input capacitance on XL1 and XL2 pads | | 4 | | pF |
| $t_{START\_LFXO}$ | Startup time for 32.768 kHz crystal oscillator | | 0.25 | | s |
| $V_{AMP\_IN\_XO\_LOW}$ | Peak to peak amplitude for external low swing clock. Input signal must not swing outside supply rails. | 200 | | 1000 | mV |

### 5.4.4.5 32.768 kHz synthesized from HFCLK (LFSYNT)

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $f_{NOM\_LFSYNT}$ | Nominal frequency | | 32.768 | | kHz |
| $f_{TOL\_LFSYNT}$ | Frequency tolerance in addition to HFLCK tolerance[12] | | 8 | | ppm |
| $t_{START\_LFSYNT}$ | Startup time for synthesized 32.768 kHz | | 100 | | us |

[11] Constant temperature within ±0.5 °C and calibration performed at least every 8 seconds, defined as 3 sigma

[12] Frequency tolerance will be derived from the HFCLK source clock plus the LFSYNT tolerance

# 6 Peripherals

## 6.1 Peripheral interface

Peripherals are controlled by the CPU by writing to configuration registers and task registers. Peripheral events are indicated to the CPU by event registers and interrupts if they are configured for a given event.

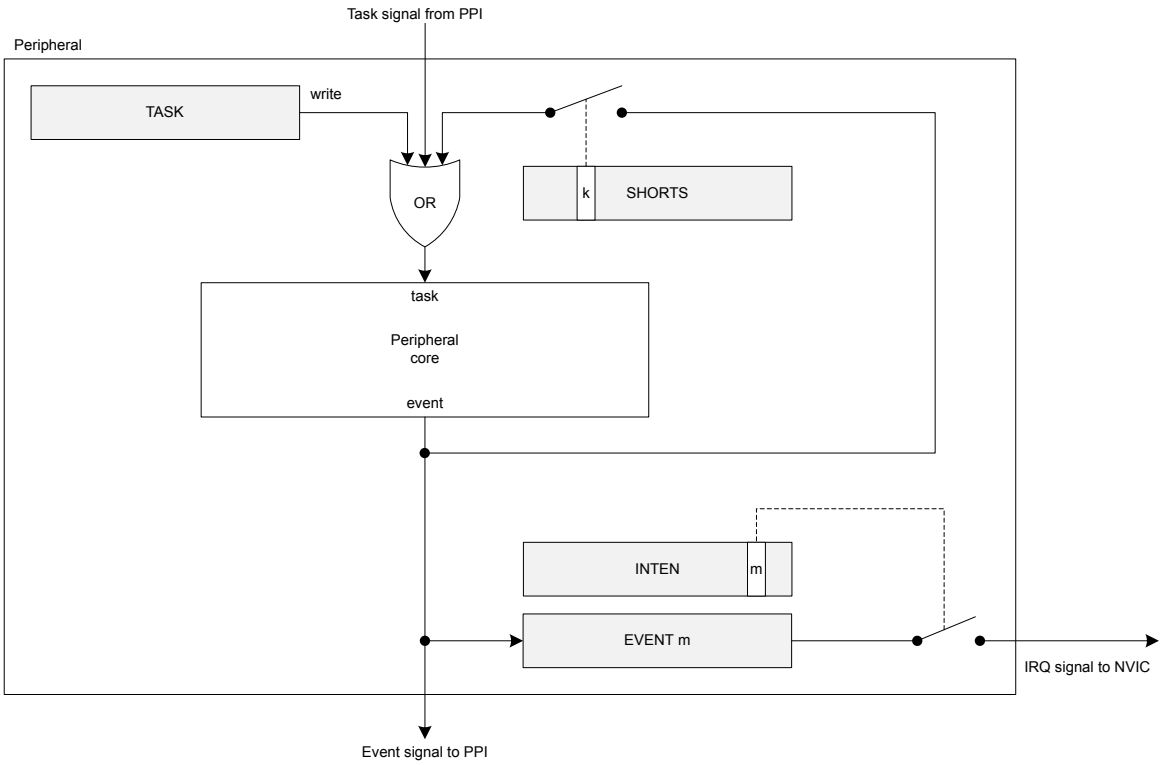



*Figure 21: Tasks, events, shortcuts, and interrupts*

### 6.1.1 Peripheral ID

Every peripheral is assigned a fixed block of 0x1000 bytes of address space, which is equal to 1024 x 32 bit registers.

See Instantiation on page 18 for more information about which peripherals are available and where they are located in the address map.

There is a direct relationship between peripheral ID and base address. For example, a peripheral with base address 0x40000000 is assigned ID=0, a peripheral with base address 0x40001000 is assigned ID=1, and a peripheral with base address 0x4001F000 is assigned ID=31.

Peripherals may share the same ID, which may impose one or more of the following limitations:

- Some peripherals share some registers or other common resources.
- Operation is mutually exclusive. Only one of the peripherals can be used at a time.
- Switching from one peripheral to another must follow a specific pattern (disable the first, then enable the second peripheral).

### 6.1.2 Peripherals with shared ID

In general (with the exception of ID 0), peripherals sharing an ID and base address may not be used simultaneously. The user can only enable one peripheral at the time on this specific ID.

When switching between two peripherals sharing an ID, the user should do the following to prevent unwanted behavior:

- Disable the previously used peripheral.
- Remove any programmable peripheral interconnect (PPI) connections set up for the peripheral that is being disabled.
- Clear all bits in the INTEN register, i.e. INTENCLR = 0xFFFFFFFF.
- Explicitly configure the peripheral that you are about to enable and do not rely on configuration values that may be inherited from the peripheral that was disabled.
- Enable the now configured peripheral.

See which peripherals are sharing ID in Instantiation on page 18.

### 6.1.3 Peripheral registers

Most peripherals feature an ENABLE register. Unless otherwise specified in the relevant chapter, the peripheral registers (in particular the PSEL registers) must be configured before enabling the peripheral.

Note that the peripheral must be enabled before tasks and events can be used.

### 6.1.4 Bit set and clear

Registers with multiple single-bit bit fields may implement the set-and-clear pattern. This pattern enables firmware to set and clear individual bits in a register without having to perform a read-modify-write operation on the main register.

This pattern is implemented using three consecutive addresses in the register map, where the main register is followed by dedicated SET and CLR registers (in that exact order).

The SET register is used to set individual bits in the main register while the CLR register is used to clear individual bits in the main register. Writing `1` to a bit in SET or CLR register will set or clear the same bit in the main register respectively. Writing `0` to a bit in SET or CLR register has no effect. Reading the SET or CLR register returns the value of the main register.

**Note:** The main register may not be visible and hence not directly accessible in all cases.

### 6.1.5 Tasks

Tasks are used to trigger actions in a peripheral, for example to start a particular behavior. A peripheral can implement multiple tasks with each task having a separate register in that peripheral's task register group.

A task is triggered when firmware writes `1` to the task register, or when the peripheral itself or another peripheral toggles the corresponding task signal. See Tasks, events, shortcuts, and interrupts on page 74.

### 6.1.6 Events

Events are used to notify peripherals and the CPU about events that have happened, for example a state change in a peripheral. A peripheral may generate multiple events with each event having a separate register in that peripheral's event register group.

An event is generated when the peripheral itself toggles the corresponding event signal, and the event register is updated to reflect that the event has been generated. See Tasks, events, shortcuts, and interrupts on page 74. An event register is only cleared when firmware writes `0` to it.

Events can be generated by the peripheral even when the event register is set to `1`.

## 6.1.7 Shortcuts

A shortcut is a direct connection between an event and a task within the same peripheral. If a shortcut is enabled, the associated task is automatically triggered when its associated event is generated.

Using a shortcut is the equivalent to making the same connection outside the peripheral and through the PPI. However, the propagation delay through the shortcut is usually shorter than the propagation delay through the PPI.

Shortcuts are predefined, which means their connections cannot be configured by firmware. Each shortcut can be individually enabled or disabled through the shortcut register, one bit per shortcut, giving a maximum of 32 shortcuts for each peripheral.

## 6.1.8 Interrupts

All peripherals support interrupts. Interrupts are generated by events.

A peripheral only occupies one interrupt, and the interrupt number follows the peripheral ID. For example, the peripheral with ID=4 is connected to interrupt number 4 in the nested vectored interrupt controller (NVIC).

Using the INTEN, INTENSET and INTENCLR registers, every event generated by a peripheral can be configured to generate that peripheral's interrupt. Multiple events can be enabled to generate interrupts simultaneously. To resolve the correct interrupt source, the event registers in the event group of peripheral registers will indicate the source.

Some peripherals implement only INTENSET and INTENCLR registers, and the INTEN register is not available on those peripherals. See the individual peripheral chapters for details. In all cases, reading back the INTENSET or INTENCLR register returns the same information as in INTEN.

Each event implemented in the peripheral is associated with a specific bit position in the INTEN, INTENSET and INTENCLR registers.

The relationship between tasks, events, shortcuts, and interrupts is shown in Tasks, events, shortcuts, and interrupts on page 74.

### Interrupt clearing

Clearing an interrupt by writing `0` to an event register, or disabling an interrupt using the INTENCLR register, can take up to four CPU clock cycles to take effect. This means that an interrupt may reoccur immediatelly, even if a new event has not come, if the program exits an interrupt handler after the interrupt is cleared or disabled but before four clock cycles have passed.

> **Note:** To avoid an interrupt reoccurring before a new event has come, the program should perform a read from one of the peripheral registers. For example, the event register that has been cleared, or the INTENCLR register that has been used to disable the interrupt. This will cause a one to three-cycle delay and ensure the interrupt is cleared before exiting the interrupt handler.

Care should be taken to ensure the compiler does not remove the read operation as an optimization. If the program can guarantee a four-cycle delay after event being cleared or interrupt disabled in any other way, then a read of a register is not required.

# 6.2 AAR — Accelerated address resolver

Accelerated address resolver is a cryptographic support function for implementing the Resolvable Private Address Resolution Procedure described in the *Bluetooth Core specification* v4.0. Resolvable Private Address generation should be achieved using ECB and is not supported by AAR.

The procedure allows two devices that share a secret key to generate and resolve a hash based on their device address. The AAR block enables real-time address resolution on incoming packets when configured as described in this chapter. This allows real-time packet filtering (whitelisting) using a list of known shared keys (Identity Resolving Keys (IRK) in *Bluetooth*).

## 6.2.1 EasyDMA

The AAR implements EasyDMA for reading and writing to the RAM. The EasyDMA will have finished accessing the RAM when the END, RESOLVED, and NOTRESOLVED events are generated.

If the IRKPTR on page 82, ADDRPTR on page 82, and the SCRATCHPTR on page 82 is not pointing to the Data RAM region, an EasyDMA transfer may result in a HardFault or RAM corruption. See Memory on page 16 for more information about the different memory regions.

## 6.2.2 Resolving a resolvable address

As per *Bluetooth* specification, a private resolvable address is composed of six bytes.

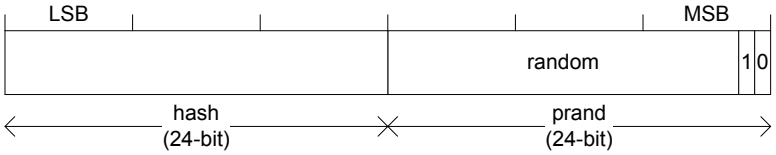



*Figure 22: Resolvable address*

To resolve an address the register ADDRPTR on page 82 must point to the start of the packet. The resolver is started by triggering the START task. A RESOLVED event is generated when the AAR manages to resolve the address using one of the Identity Resolving Keys (IRK) found in the IRK data structure. The AAR will use the IRK specified in the register IRK0 to IRK15 starting from IRK0. The register NIRK on page 81 specifies how many IRKs should be used. The AAR module will generate a NOTRESOLVED event if it is not able to resolve the address using the specified list of IRKs.

The AAR will go through the list of available IRKs in the IRK data structure and for each IRK try to resolve the address according to the Resolvable Private Address Resolution Procedure described in the *Bluetooth Core specification* v4.0 [Vol 3] chapter 10.8.2.3. The time it takes to resolve an address varies due to the location in the list of the resolvable address. The resolution time will also be affected by RAM accesses performed by other peripherals and the CPU. See the Electrical specifications for more information about resolution time.

The AAR only compares the received address to those programmed in the module without checking the address type.

The AAR will stop as soon as it has managed to resolve the address, or after trying to resolve the address using NIRK number of IRKs from the IRK data structure. The AAR will generate an END event after it has stopped.

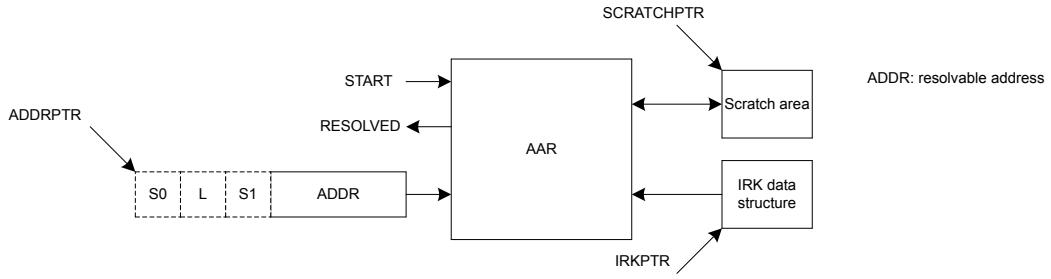



*Figure 23: Address resolution with packet preloaded into RAM*

## 6.2.3 Use case example for chaining RADIO packet reception with address resolution using AAR

The AAR may be started as soon as the 6 bytes required by the AAR have been received by the RADIO and stored in RAM. The ADDRPTR pointer must point to the start of packet.

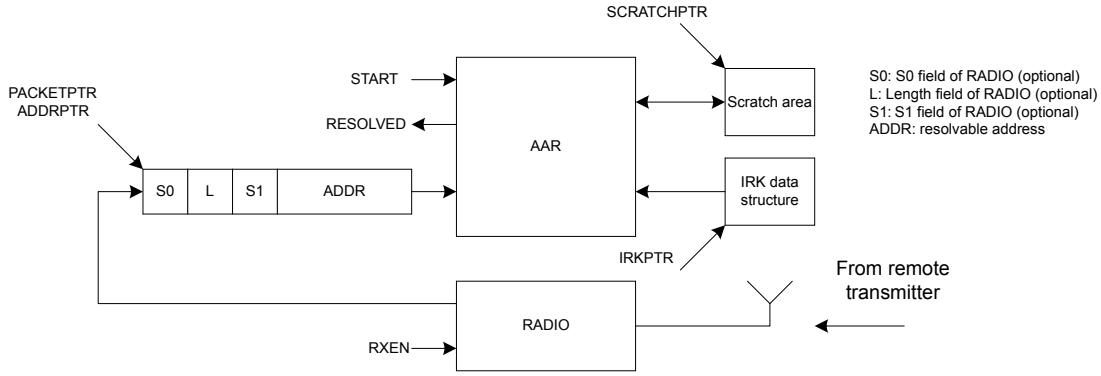



*Figure 24: Address resolution with packet loaded into RAM by the RADIO*

## 6.2.4 IRK data structure

The IRK data structure is located in RAM at the memory location specified by the IRKPTR register.

| Property | Address offset | Description |
|---|---|---|
| IRK0 | 0 | IRK number 0 (16 - byte) |
| IRK1 | 16 | IRK number 1 (16 - byte) |
| .. | .. | .. |
| IRK15 | 240 | IRK number 15 (16 - byte) |

*Table 20: IRK data structure overview*

## 6.2.5 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x4000F000 | AAR | AAR | Accelerated address resolver | |

*Table 21: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_START | 0x000 | Start resolving addresses based on IRKs specified in the IRK data structure |
| TASKS_STOP | 0x008 | Stop resolving addresses |
| EVENTS_END | 0x100 | Address resolution procedure complete |

| Register | Offset | Description |
|---|---|---|
| EVENTS_RESOLVED | 0x104 | Address resolved |
| EVENTS_NOTRESOLVED | 0x108 | Address not resolved |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| STATUS | 0x400 | Resolution status |
| ENABLE | 0x500 | Enable AAR |
| NIRK | 0x504 | Number of IRKs |
| IRKPTR | 0x508 | Pointer to IRK data structure |
| ADDRPTR | 0x510 | Pointer to the resolvable address |
| SCRATCHPTR | 0x514 | Pointer to data area used for temporary storage |

*Table 22: Register overview*

### 6.2.5.1 TASKS_START

Address offset: 0x000

Start resolving addresses based on IRKs specified in the IRK data structure

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_START | | | Start resolving addresses based on IRKs specified in the IRK data structure |
| | | | Trigger | 1 | Trigger task |

### 6.2.5.2 TASKS_STOP

Address offset: 0x008

Stop resolving addresses

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_STOP | | | Stop resolving addresses |
| | | | Trigger | 1 | Trigger task |

### 6.2.5.3 EVENTS_END

Address offset: 0x100

Address resolution procedure complete

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | EVENTS_END | | | Address resolution procedure complete |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.2.5.4 EVENTS_RESOLVED

Address offset: 0x104

Address resolved

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | EVENTS_RESOLVED | | | Address resolved |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.2.5.5 EVENTS_NOTRESOLVED

Address offset: 0x108

Address not resolved

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | EVENTS_NOTRESOLVED | | | Address not resolved |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.2.5.6 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | END | | | Write '1' to enable interrupt for event END |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | RESOLVED | | | Write '1' to enable interrupt for event RESOLVED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | NOTRESOLVED | | | Write '1' to enable interrupt for event NOTRESOLVED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

### 6.2.5.7 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | END | | | Write '1' to disable interrupt for event END |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | RESOLVED | | | Write '1' to disable interrupt for event RESOLVED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | NOTRESOLVED | | | Write '1' to disable interrupt for event NOTRESOLVED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

### 6.2.5.8 STATUS

Address offset: 0x400

Resolution status

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | STATUS | | [0..15] | The IRK that was used last time an address was resolved |

### 6.2.5.9 ENABLE

Address offset: 0x500

Enable AAR

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ENABLE | | | Enable or disable AAR |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 3 | Enable |

### 6.2.5.10 NIRK

Address offset: 0x504

Number of IRKs

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A |
| **Reset 0x00000001** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | NIRK | | [1..16] | Number of Identity root keys available in the IRK data structure |

### 6.2.5.11 IRKPTR

Address offset: 0x508

Pointer to IRK data structure

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | IRKPTR | | | Pointer to the IRK data structure |

### 6.2.5.12 ADDRPTR

Address offset: 0x510

Pointer to the resolvable address

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ADDRPTR | | | Pointer to the resolvable address (6-bytes) |

### 6.2.5.13 SCRATCHPTR

Address offset: 0x514

Pointer to data area used for temporary storage

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | SCRATCHPTR | | | Pointer to a scratch data area used for temporary storage during resolution. A space of minimum 3 bytes must be reserved. |

## 6.2.6 Electrical specification

### 6.2.6.1 AAR Electrical Specification

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{AAR}$ | Address resolution time per IRK. Total time for several IRKs is given as (1 µs + n * t_AAR), where n is the number of IRKs. (Given priority to the actual destination RAM block). | .. | .. | .. | µs |
| $t_{AAR,8}$ | Time for address resolution of 8 IRKs. (Given priority to the actual destination RAM block). | | 48 | | µs |

# 6.3 BPROT — Block protection

The mechanism for protecting non-volatile memory can be used to prevent erroneous application code from erasing or writing to protected blocks.

Non-volatile memory can be protected from erases and writes depending on the settings in the CONFIG registers. One bit in a CONFIG register represents one protected block of 4 kB. There are multiple CONFIG registers to cover the whole range of the flash. Protected regions of program memory on page 83 illustrates how the CONFIG bits map to the program memory space.

**Important:** If an erase or write to a protected block is detected, the CPU will hard fault. If an ERASEALL operation is attempted from the CPU while any block is protected, it will be blocked and the CPU will hard fault.

On reset, all the protection bits are cleared. To ensure safe operation, the first task after reset must be to set the protection bits. The only way of clearing protection bits is by resetting the device from any reset source.

The protection mechanism is turned off when in debug mode (when a debugger is connected) and the DISABLEINDEBUG register is set to disabled.

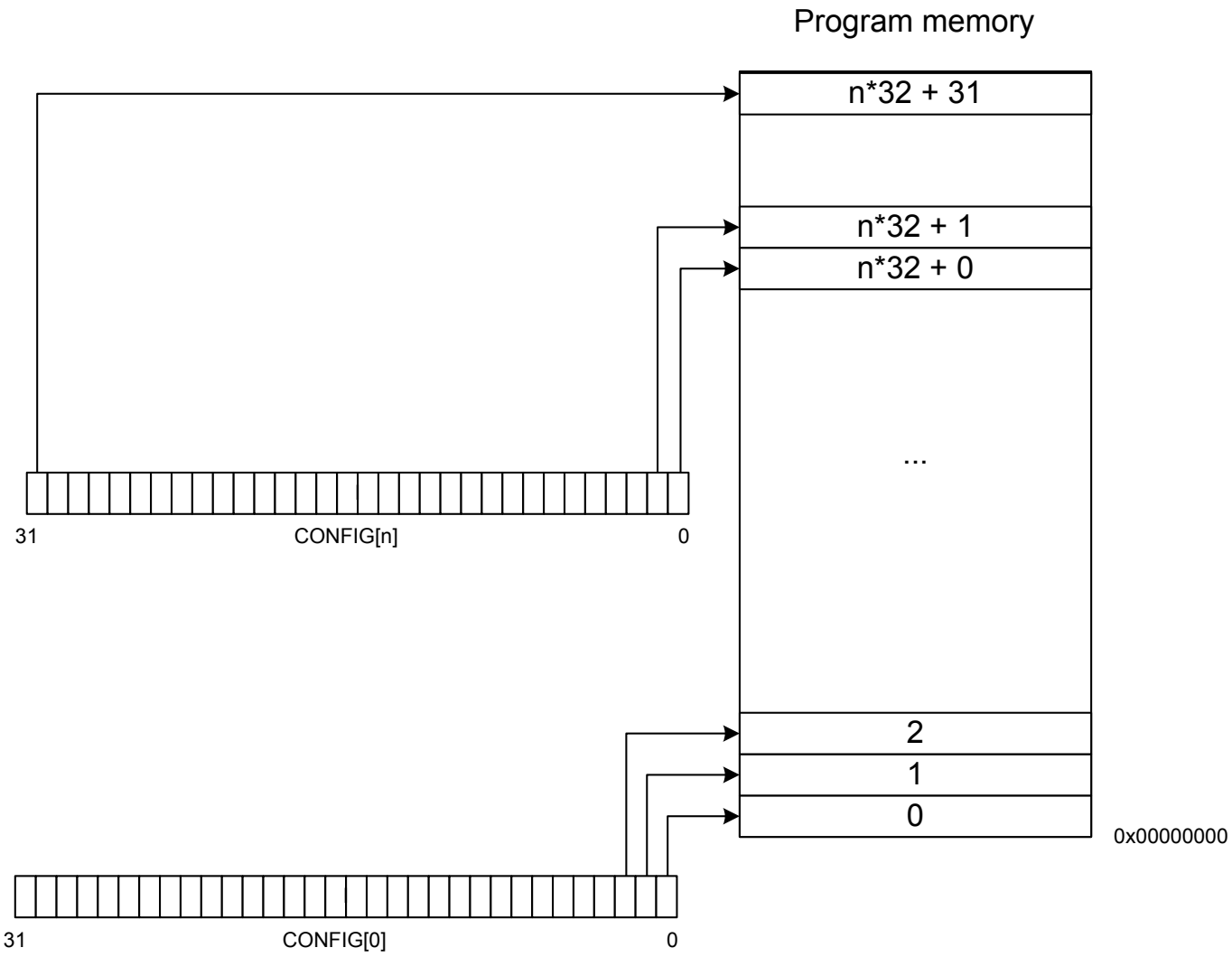



*Figure 25: Protected regions of program memory*

## 6.3.1 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40000000 | BPROT | BPROT | Block protect | |

*Table 23: Instances*

| Register | Offset | Description | |
|---|---|---|---|
| CONFIG0 | 0x600 | Block protect configuration register 0 | |
| CONFIG1 | 0x604 | Block protect configuration register 1 | |
| DISABLEINDEBUG | 0x608 | Disable protection mechanism in debug mode | |
| UNUSED0 | 0x60C | | Reserved |

*Table 24: Register overview*

### 6.3.1.1 CONFIG0

Address offset: 0x600

Block protect configuration register 0

| Bit number | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | f | e | d | c | b | a | Z | Y | X | W | V | U | T | S | R | Q | P | O | N | M | L | K | J | I | H | G | F | E | D | C | B | A |
| **Reset 0x00000000** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A-f | RW | REGION[i] (i=0..31) | | | Enable protection for region i. Write '0' has no effect. |
| | | | Disabled | 0 | Protection disabled |
| | | | Enabled | 1 | Protection enabled |

### 6.3.1.2 CONFIG1

Address offset: 0x604

Block protect configuration register 1

| Bit number | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | | | | | | | | | | | | | | | | | P | O | N | M | L | K | J | I | H | G | F | E | D | C | B | A |
| **Reset 0x00000000** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A-P | RW | REGION[i+32] (i=0..15) | | | Enable protection for region i+32. Write '0' has no effect. |
| | | | Disabled | 0 | Protection disabled |
| | | | Enabled | 1 | Protection enabled |

### 6.3.1.3 DISABLEINDEBUG

Address offset: 0x608

Disable protection mechanism in debug mode

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A |
| **Reset 0x00000001** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | DISABLEINDEBUG | | | Disable the protection mechanism for NVM regions while in debug mode. This register will only disable the protection mechanism if the device is in debug mode. |
| | | | Disabled | 1 | Disabled in debug |
| | | | Enabled | 0 | Enabled in debug |

# 6.4 CCM — AES CCM mode encryption

Cipher block chaining - message authentication code (CCM) mode is an authenticated encryption algorithm designed to provide both authentication and confidentiality during data transfer. CCM combines counter mode encryption and CBC-MAC authentication. The CCM terminology "Message authentication code (MAC)" is called the "Message integrity check (MIC)" in *Bluetooth* terminology and also in this document.

The CCM block generates an encrypted keystream that is applied to input data using the XOR operation and generates the 4 byte MIC field in one operation. The CCM and radio can be configured to work synchronously. The CCM will encrypt in time for transmission and decrypt after receiving bytes into memory from the radio. All operations can complete within the packet RX or TX time. CCM on this device is implemented according to *Bluetooth* requirements and the algorithm as defined in IETF RFC3610, and depends on the AES-128 block cipher. A description of the CCM algorithm can also be found in NIST Special Publication 800-38C. The *Bluetooth* specification describes the configuration of counter mode blocks and encryption blocks to implement compliant encryption for BLE.

The CCM block uses EasyDMA to load key, counter mode blocks (including the nonce required), and to read/write plain text and cipher text.

The AES CCM supports three operations: key-stream generation, packet encryption, and packet decryption. All these operations are done in compliance with the *Bluetooth* specification.[13]

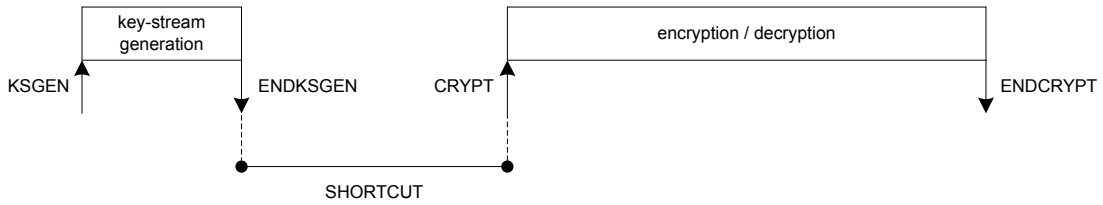



*Figure 26: Key-stream generation followed by encryption or decryption. The shortcut is optional.*

## 6.4.1 Key-steam generation

A new key-stream needs to be generated before a new packet encryption or packet decryption operation can be started.

A key-stream is generated by triggering the KSGEN task and an ENDKSGEN event will be generated when the key-stream has been generated.

Key-stream generation, packet encryption, and packet decryption operations utilize the configuration specified in the data structure pointed to by CNFPTR on page 95. It is necessary to configure this pointer and its underlying data structure, and the MODE on page 94 register before the KSGEN task is triggered.

[13] *Bluetooth* AES CCM 128 bit block encryption, see *Bluetooth* Core specification Version 4.0.

The key-stream will be stored in the AES CCM's temporary memory area, specified by the SCRATCHPTR on page 95, where it will be used in subsequent encryption and decryption operations.

For default length packets (MODE.LENGTH = Default) the size of the generated key-stream is 27 bytes. When using extended length packets (MODE.LENGTH = Extended) the MAXPACKETSIZE on page 96 register specifies the length of the key-stream to be generated. The length of the generated key-stream must be greater or equal to the length of the subsequent packet payload to be encrypted or decrypted. The maximum length of the key-stream in extended mode is 251 bytes, which means that the maximum packet payload size is 251.

If a shortcut is used between ENDKSGEN event and CRYPT task, the INPTR on page 95 pointer and the OUTPTR on page 95 pointers must also be configured before the KSGEN task is triggered.

## 6.4.2 Encryption

During packet encryption, the AES CCM will read the unencrypted packet located in RAM at the address specified in the INPTR pointer, encrypt the packet and append a four byte long Message Integrity Check (MIC) field to the packet.

Encryption is started by triggering the CRYPT task with the MODE on page 94 register set to ENCRYPTION. An ENDCRYPT event will be generated when packet encryption is completed

The AES CCM will also modify the length field of the packet to adjust for the appended MIC field, that is, add four bytes to the length, and store the resulting packet back into RAM at the address specified in the OUTPTR on page 95 pointer, see Encryption on page 86.

Empty packets (length field is set to 0) will not be encrypted but instead moved unmodified through the AES CCM.

The CCM supports different widths of the LENGTH field in the data structure for encrypted packets. This is configured in the MODE on page 94 register.

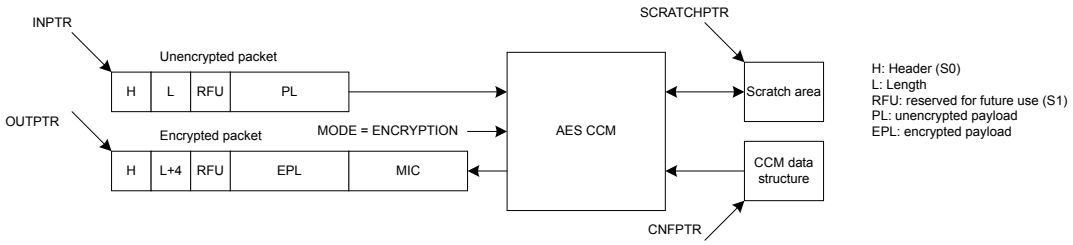



*Figure 27: Encryption*

## 6.4.3 Decryption

During packet decryption, the AES CCM will read the encrypted packet located in RAM at the address specified in the INPTR pointer, decrypt the packet, authenticate the packet's MIC field and generate the appropriate MIC status.

Decryption is started by triggering the CRYPT task with the MODE on page 94 register set to DECRYPTION. An ENDCRYPT event will be generated when packet decryption is completed

The AES CCM will also modify the length field of the packet to adjust for the MIC field, that is, subtract four bytes from the length, and then store the decrypted packet into RAM at the address pointed to by the OUTPTR pointer, see Decryption on page 87.

The CCM is only able to decrypt packet payloads that are at least 5 bytes long, that is, 1 byte or more encrypted payload (EPL) and 4 bytes of MIC. The CCM will therefore generate a MIC error for packets where the length field is set to 1, 2, 3 or 4.

Empty packets (length field is set to 0) will not be decrypted but instead moved unmodified through the AES CCM, these packets will always pass the MIC check.

The CCM supports different widths of the LENGTH field in the data structure for decrypted packets. This is configured in the MODE on page 94 register.

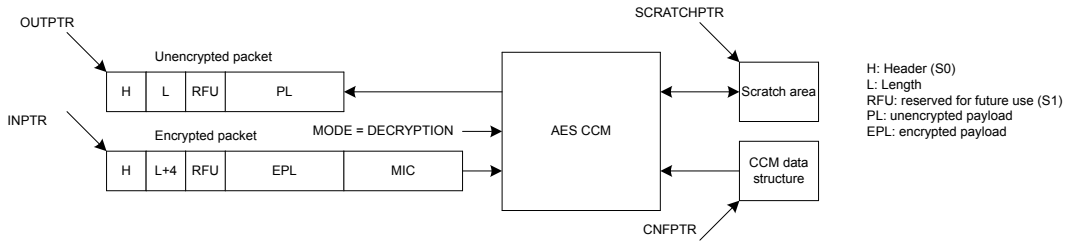



*Figure 28: Decryption*

## 6.4.4 AES CCM and RADIO concurrent operation

The CCM module is able to encrypt/decrypt data synchronously to data being transmitted or received on the radio.

In order for the CCM module to run synchronously with the radio, the data rate setting in the MODE on page 94 register needs to match the radio data rate. The settings in this register apply whenever either the KSGEN or CRYPT tasks are triggered.

The data rate setting of the MODE on page 94 register can also be overridden on-the-fly during an ongoing encrypt/decrypt operation by the contents of the RATEOVERRIDE on page 96 register. The data rate setting in this register applies whenever the RATEOVERRIDE task is triggered. This feature can be useful in cases where the radio data rate is changed during an ongoing packet transaction.

## 6.4.5 Encrypting packets on-the-fly in radio transmit mode

When the AES CCM is encrypting a packet on-the-fly at the same time as the radio is transmitting it, the radio must read the encrypted packet from the same memory location as the AES CCM is writing to.

The OUTPTR on page 95 pointer in the AES CCM must therefore point to the same memory location as the PACKETPTR pointer in the radio, see Configuration of on-the-fly encryption on page 87.

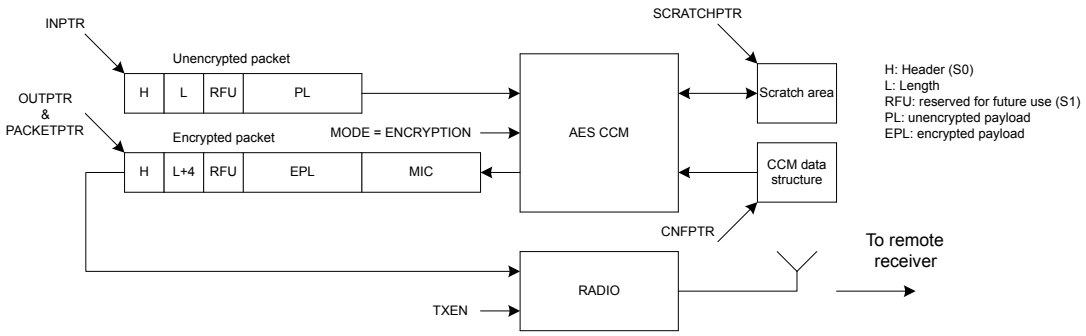



*Figure 29: Configuration of on-the-fly encryption*

In order to match the RADIO's timing, the KSGEN task must be triggered early enough to allow the key-stream generation to complete before the encryption of the packet shall start.

For short packets (MODE.LENGTH = Default) the KSGEN task must be triggered no later than when the START task in the RADIO is triggered. In addition the shortcut between the ENDKSGEN event and the CRYPT task must be enabled. This use-case is illustrated in On-the-fly encryption of short packets (MODE.LENGTH = Default) using a PPI connection on page 88 using a PPI connection between the READY event in the RADIO and the KSGEN task in the AES CCM.

For long packets (MODE.LENGTH = Extended) the key-stream generation will need to be started even earlier, for example at the time when the TXEN task in the RADIO is triggered.

**Important:** Refer to Timing specification on page 97 for information about the time needed for generating a key-stream.

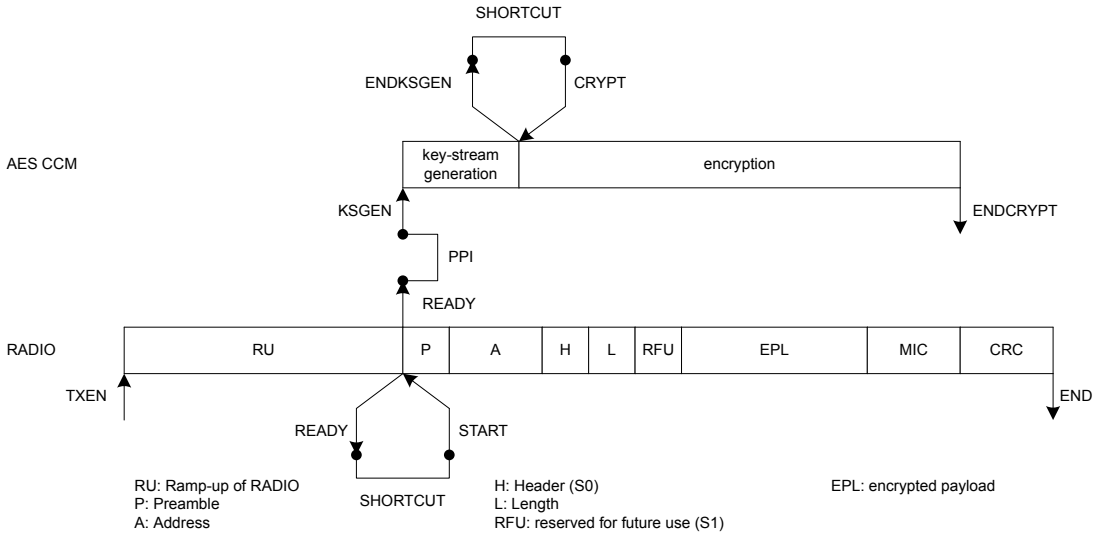



*Figure 30: On-the-fly encryption of short packets (MODE.LENGTH = Default) using a PPI connection*

## 6.4.6 Decrypting packets on-the-fly in radio receive mode

When the AES CCM is decrypting a packet on-the-fly at the same time as the RADIO is receiving it, the AES CCM must read the encrypted packet from the same memory location as the RADIO is writing to.

The INPTR on page 95 pointer in the AES CCM must therefore point to the same memory location as the PACKETPTR pointer in the RADIO, see Configuration of on-the-fly decryption on page 88.

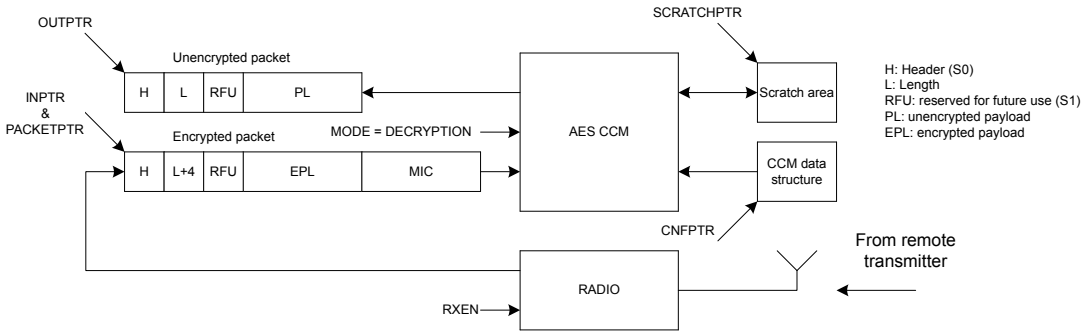



*Figure 31: Configuration of on-the-fly decryption*

In order to match the RADIO's timing, the KSGEN task must be triggered early enough to allow the key-stream generation to complete before the decryption of the packet shall start.

For short packets (MODE.LENGTH = Default) the KSGEN task must be triggered no later than when the START task in the RADIO is triggered. In addition, the CRYPT task must be triggered no earlier than when the ADDRESS event is generated by the RADIO.

If the CRYPT task is triggered exactly at the same time as the ADDRESS event is generated by the RADIO, the AES CCM will guarantee that the decryption is completed no later than when the END event in the RADIO is generated.

This use-case is illustrated in On-the-fly decryption of short packets (MODE.LENGTH = Default) using a PPI connection on page 89 using a PPI connection between the ADDRESS event in the RADIO and the CRYPT task in the AES CCM. The KSGEN task is triggered from the READY event in the RADIO through a PPI connection.

For long packets (MODE.LENGTH = Extended) the key-stream generation will need to be started even earlier, for example at the time when the RXEN task in the RADIO is triggered.

**Important:** Refer to Timing specification on page 97 for information about the time needed for generating a key-stream.

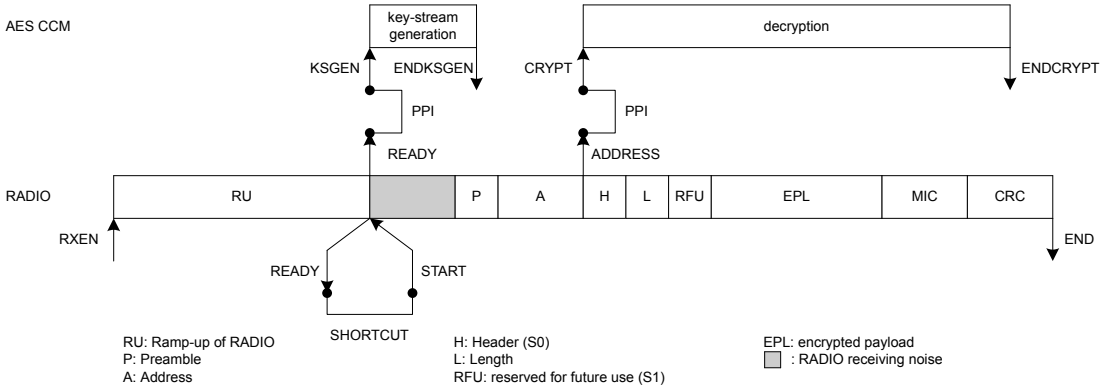



*Figure 32: On-the-fly decryption of short packets (MODE.LENGTH = Default) using a PPI connection*

## 6.4.7 CCM data structure

The CCM data structure is located in Data RAM at the memory location specified by the CNFPTR pointer register.

| Property | Address offset | Description |
|---|---|---|
| KEY | 0 | 16 byte AES key |
| PKTCTR | 16 | Octet0 (LSO) of packet counter |
| | 17 | Octet1 of packet counter |
| | 18 | Octet2 of packet counter |
| | 19 | Octet3 of packet counter |
| | 20 | Bit 6 – Bit 0: Octet4 (7 most significant bits of packet counter, with Bit 6 being the most significant bit) Bit7: Ignored |
| | 21 | Ignored |
| | 22 | Ignored |
| | 23 | Ignored |
| | 24 | Bit 0: Direction bit Bit 7 – Bit 1: Zero padded |
| IV | 25 | 8 byte initialization vector (IV) Octet0 (LSO) of IV, Octet1 of IV, … , Octet7 (MSO) of IV |

*Table 25: CCM data structure overview*

The NONCE vector (as specified by the *Bluetooth* Core Specification) will be generated by hardware based on the information specified in the CCM data structure from CCM data structure overview on page 89 .

| Property | Address offset | Description |
|---|---|---|
| HEADER | 0 | Packet Header |
| LENGTH | 1 | Number of bytes in unencrypted payload |
| RFU | 2 | Reserved Future Use |
| PAYLOAD | 3 | Unencrypted payload |

*Table 26: Data structure for unencrypted packet*

| Property | Address offset | Description |
|---|---|---|
| HEADER | 0 | Packet Header |
| LENGTH | 1 | Number of bytes in encrypted payload including length of MIC<br>**Important:** LENGTH will be 0 for empty packets since the MIC is not added to empty packets |
| RFU | 2 | Reserved Future Use |
| PAYLOAD | 3 | Encrypted payload |
| MIC | 3 + payload length | ENCRYPT: 4 bytes encrypted MIC<br>**Important:** MIC is not added to empty packets |

*Table 27: Data structure for encrypted packet*

## 6.4.8 EasyDMA and ERROR event

The CCM implements an EasyDMA mechanism for reading and writing to the RAM.

In cases where the CPU and other EasyDMA enabled peripherals are accessing the same RAM block at the same time, a high level of bus collisions may cause too slow operation for correct on the fly encryption. In this case the ERROR event will be generated.

The EasyDMA will have finished accessing the RAM when the ENDKSGEN and ENDCRYPT events are generated.

If the CNFPTR, SCRATCHPTR, INPTR and the OUTPTR are not pointing to the Data RAM region, an EasyDMA transfer may result in a HardFault or RAM corruption. See Memory on page 16 for more information about the different memory regions.

## 6.4.9 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x4000F000 | CCM | CCM | AES CCM mode encryption | |

*Table 28: Instances*

| Register | Offset | Description | |
|---|---|---|---|
| TASKS_KSGEN | 0x000 | Start generation of key-stream. This operation will stop by itself when completed. | |
| TASKS_CRYPT | 0x004 | Start encryption/decryption. This operation will stop by itself when completed. | |
| TASKS_STOP | 0x008 | Stop encryption/decryption | |
| TASKS_RATEOVERRIDE | 0x00C | Override DATARATE setting in MODE register with the contents of the RATEOVERRIDE register for any ongoing encryption/decryption | |
| EVENTS_ENDKSGEN | 0x100 | Key-stream generation complete | |
| EVENTS_ENDCRYPT | 0x104 | Encrypt/decrypt complete | |
| EVENTS_ERROR | 0x108 | CCM error event | Deprecated |
| SHORTS | 0x200 | Shortcuts between local events and tasks | |
| INTENSET | 0x304 | Enable interrupt | |
| INTENCLR | 0x308 | Disable interrupt | |
| MICSTATUS | 0x400 | MIC check result | |
| ENABLE | 0x500 | Enable | |
| MODE | 0x504 | Operation mode | |
| CNFPTR | 0x508 | Pointer to data structure holding AES key and NONCE vector | |
| INPTR | 0x50C | Input pointer | |
| OUTPTR | 0x510 | Output pointer | |
| SCRATCHPTR | 0x514 | Pointer to data area used for temporary storage | |

| Register | Offset | Description |
|---|---|---|
| MAXPACKETSIZE | 0x518 | Length of key-stream generated when MODE.LENGTH = Extended. |
| RATEOVERRIDE | 0x51C | Data rate override setting. |

*Table 29: Register overview*

### 6.4.9.1 TASKS_KSGEN

Address offset: 0x000

Start generation of key-stream. This operation will stop by itself when completed.

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_KSGEN | | | Start generation of key-stream. This operation will stop by itself when completed. |
| | | | Trigger | 1 | Trigger task |

### 6.4.9.2 TASKS_CRYPT

Address offset: 0x004

Start encryption/decryption. This operation will stop by itself when completed.

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_CRYPT | | | Start encryption/decryption. This operation will stop by itself when completed. |
| | | | Trigger | 1 | Trigger task |

### 6.4.9.3 TASKS_STOP

Address offset: 0x008

Stop encryption/decryption

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_STOP | | | Stop encryption/decryption |
| | | | Trigger | 1 | Trigger task |

### 6.4.9.4 TASKS_RATEOVERRIDE

Address offset: 0x00C

Override DATARATE setting in MODE register with the contents of the RATEOVERRIDE register for any ongoing encryption/decryption

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_RATEOVERRIDE | | | Override DATARATE setting in MODE register with the contents of the RATEOVERRIDE register for any ongoing encryption/decryption |
| | | | Trigger | 1 | Trigger task |

### 6.4.9.5 EVENTS_ENDKSGEN

Address offset: 0x100

Key-stream generation complete

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_ENDKSGEN | | | Key-stream generation complete |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.4.9.6 EVENTS_ENDCRYPT

Address offset: 0x104

Encrypt/decrypt complete

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_ENDCRYPT | | | Encrypt/decrypt complete |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.4.9.7 EVENTS_ERROR ( Deprecated )

Address offset: 0x108

CCM error event

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_ERROR | | | CCM error event Deprecated |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.4.9.8 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ENDKSGEN_CRYPT | | | Shortcut between event ENDKSGEN and task CRYPT |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |

## 6.4.9.9 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ENDKSGEN | | | Write '1' to enable interrupt for event ENDKSGEN |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | ENDCRYPT | | | Write '1' to enable interrupt for event ENDCRYPT |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | ERROR | | | Write '1' to enable interrupt for event ERROR Deprecated |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.4.9.10 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ENDKSGEN | | | Write '1' to disable interrupt for event ENDKSGEN |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | ENDCRYPT | | | Write '1' to disable interrupt for event ENDCRYPT |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | ERROR | | | Write '1' to disable interrupt for event ERROR Deprecated |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

### 6.4.9.11 MICSTATUS

Address offset: 0x400

MIC check result

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | MICSTATUS | | | The result of the MIC check performed during the previous decryption operation |
| | | | CheckFailed | 0 | MIC check failed |
| | | | CheckPassed | 1 | MIC check passed |

### 6.4.9.12 ENABLE

Address offset: 0x500

Enable

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ENABLE | | | Enable or disable CCM |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 2 | Enable |

### 6.4.9.13 MODE

Address offset: 0x504

Operation mode

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B B A |
| **Reset 0x00000001** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | MODE | | | The mode of operation to be used. The settings in this register apply whenever either the KSGEN or CRYPT tasks are triggered. |
| | | | Encryption | 0 | AES CCM packet encryption mode |
| | | | Decryption | 1 | AES CCM packet decryption mode |
| B | RW | DATARATE | | | Radio data rate that the CCM shall run synchronous with |
| | | | 1Mbit | 0 | 1 Mbps |
| | | | 2Mbit | 1 | 2 Mbps |
| | | | 125Kbps | 2 | 125 Kbps |
| | | | 500Kbps | 3 | 500 Kbps |
| C | RW | LENGTH | | | Packet length configuration |
| | | | Default | 0 | Default length. Effective length of LENGTH field in encrypted/decrypted packet is 5 bits. A key-stream for packet payloads up to 27 bytes will be generated. |

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | C B B A |
| **Reset 0x00000001** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1** |
| ID | Acce Field | Value ID | Value | Description |
| | | Extended | 1 | Extended length. Effective length of LENGTH field in encrypted/decrypted packet is 8 bits. A key-stream for packet payloads up to MAXPACKETSIZE bytes will be generated. |

## 6.4.9.14 CNFPTR

Address offset: 0x508

Pointer to data structure holding AES key and NONCE vector

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce Field | Value ID | Value | Description |
| A | RW CNFPTR | | | Pointer to the data structure holding the AES key and the CCM NONCE vector (see Table 1 CCM data structure overview) |

## 6.4.9.15 INPTR

Address offset: 0x50C

Input pointer

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce Field | Value ID | Value | Description |
| A | RW INPTR | | | Input pointer |

## 6.4.9.16 OUTPTR

Address offset: 0x510

Output pointer

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce Field | Value ID | Value | Description |
| A | RW OUTPTR | | | Output pointer |

## 6.4.9.17 SCRATCHPTR

Address offset: 0x514

Pointer to data area used for temporary storage

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | SCRATCHPTR | | | Pointer to a scratch data area used for temporary storage during key-stream generation, MIC generation and encryption/decryption.<br>The scratch area is used for temporary storage of data during key-stream generation and encryption.<br>When MODE.LENGTH = Default, a space of 43 bytes is required for this temporary storage. MODE.LENGTH = Extended (16 + MAXPACKETSIZE) bytes of storage is required. |

## 6.4.9.18 MAXPACKETSIZE

Address offset: 0x518

Length of key-stream generated when MODE.LENGTH = Extended.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A |
| **Reset 0x000000FB** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | MAXPACKETSIZE | | [0x001B..0x00FB] | Length of key-stream generated when MODE.LENGTH = Extended. This value must be greater or equal to the subsequent packet payload to be encrypted/decrypted. |

## 6.4.9.19 RATEOVERRIDE

Address offset: 0x51C

Data rate override setting.

Override value to be used instead of the setting of MODE.DATARATE. This override value applies when the RATEOVERRIDE task is triggered.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | RATEOVERRIDE | | | Data rate override setting. |
| | | | 1Mbit | 0 | 1 Mbps |
| | | | 2Mbit | 1 | 2 Mbps |
| | | | 125Kbps | 2 | 125 Kbps |
| | | | 500Kbps | 3 | 500 Kbps |

## 6.4.10 Electrical specification

### 6.4.10.1 Timing specification

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{kgen}$ | Time needed for key-stream generation (given priority access to destination RAM block). | .. | .. | .. | µs |

# 6.5 COMP — Comparator

The comparator (COMP) compares an input voltage (VIN+) against a second input voltage (VIN-). VIN+ can be derived either from an analog input pin (AIN0-AIN6) or VDD/2. VIN- can be derived from multiple sources depending on the operation mode of the comparator.

Main features of the comparator are:

- Input range from 0 V to VDD
- Single-ended mode
  - Fully flexible hysteresis using a 64-level reference ladder
- Differential mode
  - Configurable hysteresis
- Reference inputs (VREF):
  - VDD
  - External reference from AIN0 to AIN7 (between 0 V and VDD)
  - Internal references 1.2 V, 1.8 V and 2.4 V
- Three speed/power consumption modes: low-power, normal and high-speed
- Event generation on output changes
  - UP event on VIN- > VIN+
  - DOWN event on VIN- < VIN+
  - CROSS event on VIN+ and VIN- crossing
  - READY event on core and internal reference (if used) ready

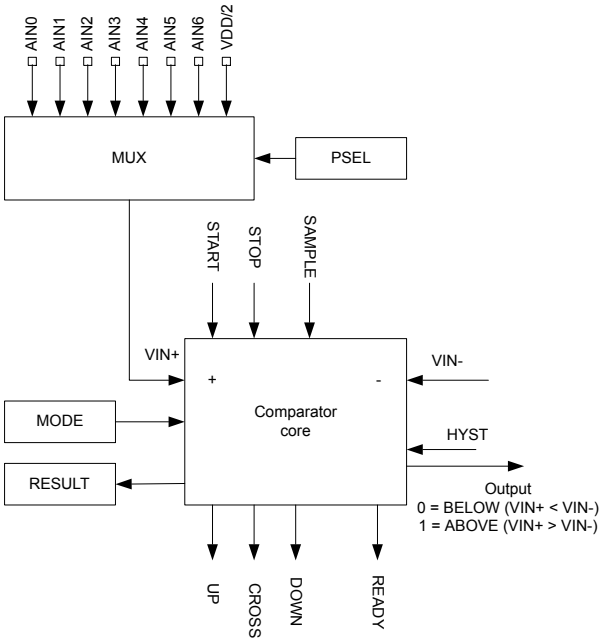



*Figure 33: Comparator overview*

Once enabled (using the ENABLE register), the comparator is started by triggering the START task and stopped by triggering the STOP task. After a start-up time of $t_{COMP,START}$, the comparator will generate a READY event to indicate that it is ready for use and that its output is correct. When the COMP module is started, events will be generated every time VIN+ crosses VIN-.

## Operation modes

The comparator can be configured to operate in two main operation modes, differential mode and single-ended mode. See the MODE register for more information. In both operation modes, the comparator can operate in different speed and power consumption modes (low-power, normal and high-speed). High-speed mode will consume more power compared to low-power mode, and low-power mode will result in slower response time compared to high-speed mode.

Use the PSEL register to select any of the AIN0-AIN6 pins (or VDD/2) as VIN+ input, irregardless of the operation mode selected for the comparator. The source of VIN- depends on which operation mode is used:

- Differential mode: Derived directly from AIN0 to AIN7
- Single-ended mode: Derived from VREF. VREF can be derived from VDD, AIN0-AIN7 or internal 1.2 V, 1.8 V and 2.4 V references.

The selected analog pins will be acquired by the comparator once it is enabled.

An optional hysteresis on VIN+ and VIN- can be enabled when the module is used in differential mode through the HYST register. In single-ended mode, VUP and VDOWN thresholds can be set to implement a hysteresis using the reference ladder (see Comparator in single-ended mode on page 100). This hysteresis is in the order of magnitude of 30 mV, and shall prevent noise on the signal to create unwanted events. See Hysteresis example where VIN+ starts below VUP on page 101 for illustration of the effect of an active hysteresis on a noisy input signal.

An upward crossing will generate an UP event and a downward crossing will generate a DOWN event. The CROSS event will be generated every time there is a crossing, independent of direction.

The immediate value of the comparator can be sampled to RESULT register by triggering the SAMPLE task.

## 6.5.1 Differential mode

In differential mode, the reference input VIN- is derived directly from one of the AINx pins.

Before enabling the comparator via the ENABLE register, the following registers must be configured for the differential mode:

- PSEL
- MODE
- EXTREFSEL

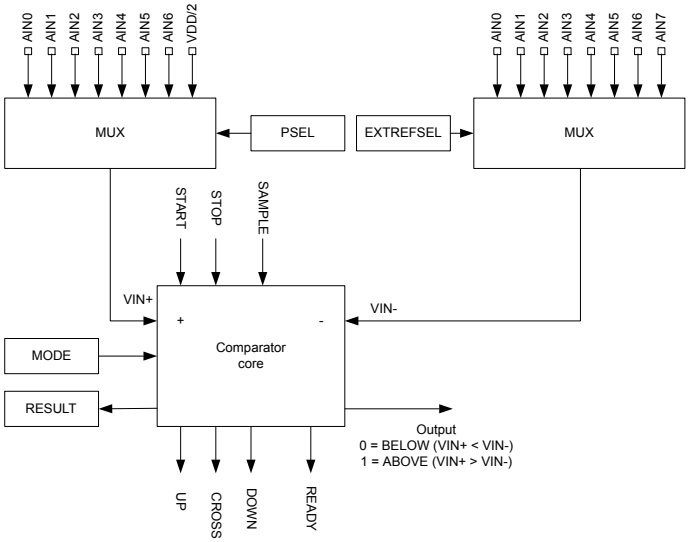



*Figure 34: Comparator in differential mode*

**Restriction:** Depending on the device, not all the analog inputs may be available for each MUX. See definitions for PSEL and EXTREFSEL for more information about which analog pins are available on a particular device.

When HYST register is turned on while in this mode, the output of the comparator (and associated events) will change from ABOVE to BELOW whenever VIN+ becomes lower than VIN- - ($V_{DIFFHYST}$ / 2). It will also change from BELOW to ABOVE whenever VIN+ becomes higher than VIN- + ($V_{DIFFHYST}$ / 2). This behavior is illustrated in Hysteresis enabled in differential mode on page 99.

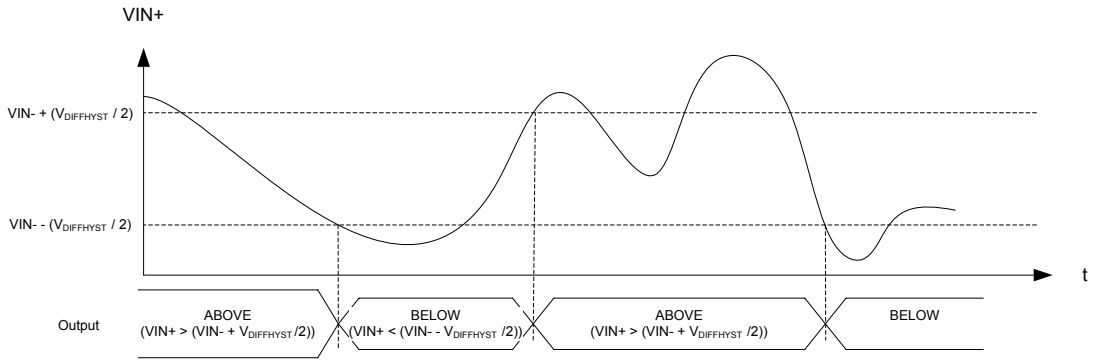



*Figure 35: Hysteresis enabled in differential mode*

## 6.5.2 Single-ended mode

In single-ended mode, VIN- is derived from the reference ladder.

Before enabling the comparator via the ENABLE register, the following registers must be configured for the single-ended mode:

- PSEL
- MODE
- REFSEL
- EXTREFSEL
- TH

The reference ladder uses the reference voltage (VREF) to derive two new voltage references, VUP and VDOWN. VUP and VDOWN are configured using THUP and THDOWN respectively in the TH register. VREF can be derived from any of the available reference sources, configured using the EXTREFSEL and REFSEL registers as illustrated in Comparator in single-ended mode on page 100. When AREF is selected in the REFSEL register, the EXTREFSEL register is used to select one of the AIN0-AIN7 analog input pins as reference input. The selected analog pins will be acquired by the comparator once it is enabled.

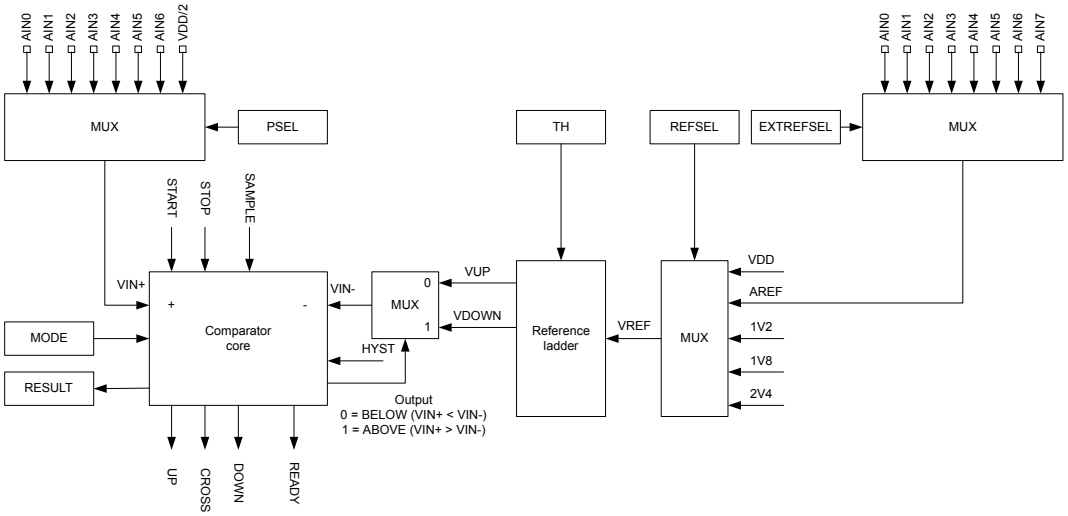



*Figure 36: Comparator in single-ended mode*

**Restriction:** Depending on the device, not all the analog inputs may be available for each MUX. See definitions for PSEL and EXTREFSEL for more information about which analog pins are available on a particular device.

When the comparator core detects that VIN+ > VIN-, i.e. ABOVE as per the RESULT register, VIN- will switch to VDOWN. When VIN+ falls below VIN- again, VIN- will be switched back to VUP. By specifying VUP larger than VDOWN, a hysteresis can be generated as illustrated in Hysteresis example where VIN+ starts below VUP on page 101 and Hysteresis example where VIN+ starts above VUP on page 101.

Writing to HYST has no effect in single-ended mode, and the content of this register is ignored.

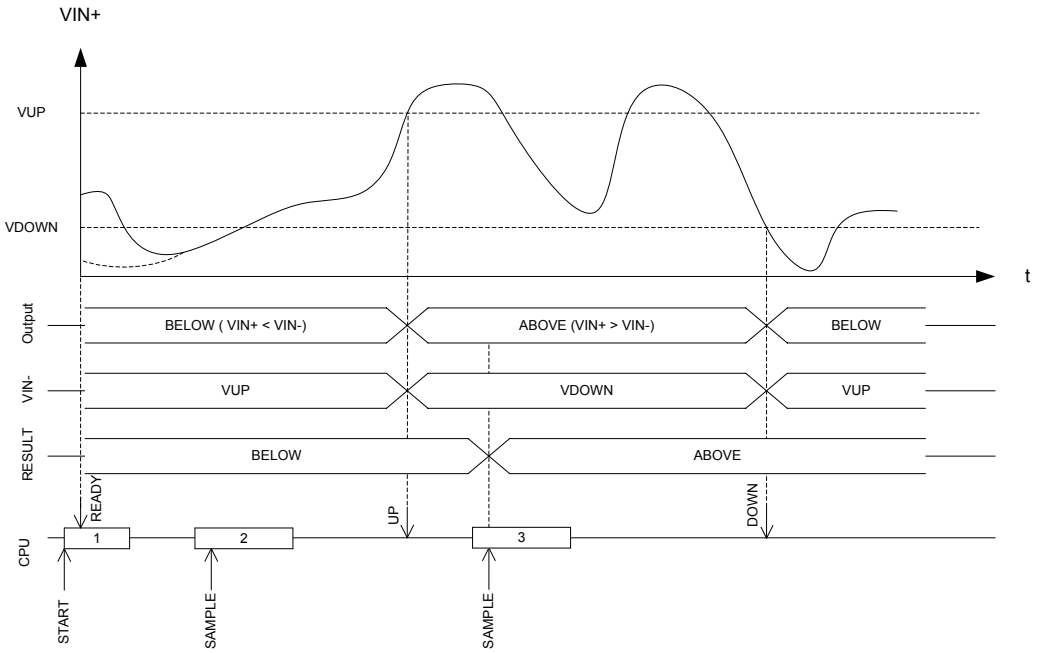



*Figure 37: Hysteresis example where VIN+ starts below VUP*

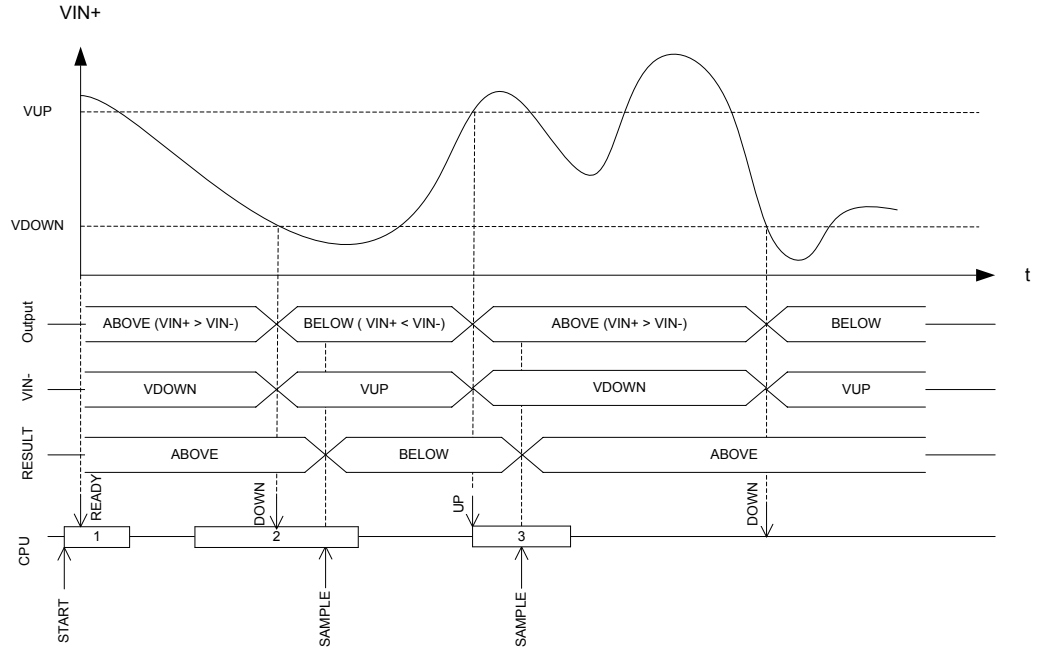



*Figure 38: Hysteresis example where VIN+ starts above VUP*

## 6.5.3 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40013000 | COMP | COMP | General purpose comparator | |

*Table 30: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_START | 0x000 | Start comparator |

| Register | Offset | Description |
|---|---|---|
| TASKS_STOP | 0x004 | Stop comparator |
| TASKS_SAMPLE | 0x008 | Sample comparator value |
| EVENTS_READY | 0x100 | COMP is ready and output is valid |
| EVENTS_DOWN | 0x104 | Downward crossing |
| EVENTS_UP | 0x108 | Upward crossing |
| EVENTS_CROSS | 0x10C | Downward or upward crossing |
| SHORTS | 0x200 | Shortcuts between local events and tasks |
| INTEN | 0x300 | Enable or disable interrupt |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| RESULT | 0x400 | Compare result |
| ENABLE | 0x500 | COMP enable |
| PSEL | 0x504 | Pin select |
| REFSEL | 0x508 | Reference source select for single-ended mode |
| EXTREFSEL | 0x50C | External reference select |
| TH | 0x530 | Threshold configuration for hysteresis unit |
| MODE | 0x534 | Mode configuration |
| HYST | 0x538 | Comparator hysteresis enable |

*Table 31: Register overview*

### 6.5.3.1 TASKS_START

Address offset: 0x000

Start comparator

| Bit number | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | A |
| **Reset 0x00000000** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_START | | | Start comparator |
| | | | Trigger | 1 | Trigger task |

### 6.5.3.2 TASKS_STOP

Address offset: 0x004

Stop comparator

| Bit number | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | A |
| **Reset 0x00000000** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_STOP | | | Stop comparator |
| | | | Trigger | 1 | Trigger task |

### 6.5.3.3 TASKS_SAMPLE

Address offset: 0x008

Sample comparator value

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_SAMPLE | | | Sample comparator value |
| | | | Trigger | 1 | Trigger task |

### 6.5.3.4 EVENTS_READY

Address offset: 0x100

COMP is ready and output is valid

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_READY | | | COMP is ready and output is valid |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.5.3.5 EVENTS_DOWN

Address offset: 0x104

Downward crossing

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_DOWN | | | Downward crossing |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.5.3.6 EVENTS_UP

Address offset: 0x108

Upward crossing

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_UP | | | Upward crossing |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.5.3.7 EVENTS_CROSS

Address offset: 0x10C

Downward or upward crossing

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_CROSS | | | Downward or upward crossing |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.5.3.8 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | READY_SAMPLE | | | Shortcut between event READY and task SAMPLE |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| B | RW | READY_STOP | | | Shortcut between event READY and task STOP |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| C | RW | DOWN_STOP | | | Shortcut between event DOWN and task STOP |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| D | RW | UP_STOP | | | Shortcut between event UP and task STOP |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| E | RW | CROSS_STOP | | | Shortcut between event CROSS and task STOP |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |

## 6.5.3.9 INTEN

Address offset: 0x300

Enable or disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | READY | | | Enable or disable interrupt for event READY |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| B | RW | DOWN | | | Enable or disable interrupt for event DOWN |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| C | RW | UP | | | Enable or disable interrupt for event UP |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| D | RW | CROSS | | | Enable or disable interrupt for event CROSS |
| | | | Disabled | 0 | Disable |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| | | | Enabled | 1 | Enable |

## 6.5.3.10 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | READY | | | Write '1' to enable interrupt for event READY |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | DOWN | | | Write '1' to enable interrupt for event DOWN |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | UP | | | Write '1' to enable interrupt for event UP |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | CROSS | | | Write '1' to enable interrupt for event CROSS |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.5.3.11 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | READY | | | Write '1' to disable interrupt for event READY |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | DOWN | | | Write '1' to disable interrupt for event DOWN |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | UP | | | Write '1' to disable interrupt for event UP |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| D | RW | CROSS | | | Write '1' to disable interrupt for event CROSS |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.5.3.12 RESULT

Address offset: 0x400

Compare result

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | RESULT | | | Result of last compare. Decision point SAMPLE task. |
| | | | Below | 0 | Input voltage is below the threshold (VIN+ < VIN-) |
| | | | Above | 1 | Input voltage is above the threshold (VIN+ > VIN-) |

## 6.5.3.13 ENABLE

Address offset: 0x500

COMP enable

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ENABLE | | | Enable or disable COMP |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 2 | Enable |

## 6.5.3.14 PSEL

Address offset: 0x504

Pin select

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PSEL | | | Analog pin select |
| | | | AnalogInput0 | 0 | AIN0 selected as analog input |
| | | | AnalogInput1 | 1 | AIN1 selected as analog input |
| | | | AnalogInput2 | 2 | AIN2 selected as analog input |
| | | | AnalogInput3 | 3 | AIN3 selected as analog input |
| | | | AnalogInput4 | 4 | AIN4 selected as analog input |
| | | | AnalogInput5 | 5 | AIN5 selected as analog input |
| | | | AnalogInput6 | 6 | AIN6 selected as analog input |
| | | | VddDiv2 | 7 | VDD/2 selected as analog input |

### 6.5.3.15 REFSEL

Address offset: 0x508

Reference source select for single-ended mode

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A |
| **Reset 0x00000004** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | REFSEL | | | Reference select |
| | | | Int1V2 | 0 | VREF = internal 1.2 V reference (VDD >= 1.7 V) |
| | | | Int1V8 | 1 | VREF = internal 1.8 V reference (VDD >= VREF + 0.2 V) |
| | | | Int2V4 | 2 | VREF = internal 2.4 V reference (VDD >= VREF + 0.2 V) |
| | | | VDD | 4 | VREF = VDD |
| | | | ARef | 5 | VREF = AREF (VDD >= VREF >= AREFMIN) |

### 6.5.3.16 EXTREFSEL

Address offset: 0x50C

External reference select

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EXTREFSEL | | | External analog reference select |
| | | | AnalogReference0 | 0 | Use AIN0 as external analog reference |
| | | | AnalogReference1 | 1 | Use AIN1 as external analog reference |
| | | | AnalogReference2 | 2 | Use AIN2 as external analog reference |
| | | | AnalogReference3 | 3 | Use AIN3 as external analog reference |
| | | | AnalogReference4 | 4 | Use AIN4 as external analog reference |
| | | | AnalogReference5 | 5 | Use AIN5 as external analog reference |
| | | | AnalogReference6 | 6 | Use AIN6 as external analog reference |
| | | | AnalogReference7 | 7 | Use AIN7 as external analog reference |

### 6.5.3.17 TH

Address offset: 0x530

Threshold configuration for hysteresis unit

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | B B B B B B A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | THDOWN | | [63:0] | VDOWN = (THDOWN+1)/64*VREF |
| B | RW | THUP | | [63:0] | VUP = (THUP+1)/64*VREF |

### 6.5.3.18 MODE

Address offset: 0x534

Mode configuration

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | B A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | SP | | | Speed and power modes |
| | | | Low | 0 | Low-power mode |
| | | | Normal | 1 | Normal mode |
| | | | High | 2 | High-speed mode |
| B | RW | MAIN | | | Main operation modes |
| | | | SE | 0 | Single-ended mode |
| | | | Diff | 1 | Differential mode |

### 6.5.3.19 HYST

Address offset: 0x538

Comparator hysteresis enable

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | HYST | | | Comparator hysteresis |
| | | | NoHyst | 0 | Comparator hysteresis disabled |
| | | | Hyst50mV | 1 | Comparator hysteresis enabled |

## 6.5.4 Electrical specification

### 6.5.4.1 COMP Electrical Specification

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{PROPDLY,LP}$ | Propagation delay, low-power mode[a] | | 0.6 | | µS |
| $t_{PROPDLY,N}$ | Propagation delay, normal mode[a] | | 0.2 | | µS |
| $t_{PROPDLY,HS}$ | Propagation delay, high-speed mode[a] | | 0.1 | | µS |
| $V_{DIFFHYST}$ | Optional hysteresis applied to differential input | | 30 | | mV |

[a] Propagation delay is with 10 mV overdrive.

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $V_{VDD-VREF}$ | Required difference between VDD and a selected VREF, VDD > VREF | 0.3 | | | V |
| $t_{INT\_REF,START}$ | Startup time for the internal bandgap reference | | 50 | 80 | µS |
| $E_{INT\_REF}$ | Internal bandgap reference error | -3 | | 3 | % |
| $V_{INPUTOFFSET}$ | Input offset | -10 | | 10 | mV |
| $t_{COMP,START}$ | Startup time for the comparator core | | 3 | | µS |

# 6.6 ECB — AES electronic codebook mode encryption

The AES electronic codebook mode encryption (ECB) can be used for a range of cryptographic functions like hash generation, digital signatures, and keystream generation for data encryption/decryption. The ECB encryption block supports 128 bit AES encryption (encryption only, not decryption).

AES ECB operates with EasyDMA access to system Data RAM for in-place operations on cleartext and ciphertext during encryption. ECB uses the same AES core as the CCM and AAR blocks and is an asynchronous operation which may not complete if the AES core is busy.

AES ECB features:

- 128 bit AES encryption
- Supports standard AES ECB block encryption
- Memory pointer support
- DMA data transfer

AES ECB performs a 128 bit AES block encrypt. At the STARTECB task, data and key is loaded into the algorithm by EasyDMA. When output data has been written back to memory, the ENDECB event is triggered.

AES ECB can be stopped by triggering the STOPECB task.

## 6.6.1 Shared resources

The ECB, CCM, and AAR share the same AES module. The ECB will always have lowest priority and if there is a sharing conflict during encryption, the ECB operation will be aborted and an ERRORECB event will be generated.

## 6.6.2 EasyDMA

The ECB implements an EasyDMA mechanism for reading and writing to the Data RAM. This DMA cannot access the program memory or any other parts of the memory area except RAM.

If the ECBDATAPTR is not pointing to the Data RAM region, an EasyDMA transfer may result in a HardFault or RAM corruption. See Memory on page 16 for more information about the different memory regions.

The EasyDMA will have finished accessing the Data RAM when the ENDECB or ERRORECB is generated.

## 6.6.3 ECB data structure

Input to the block encrypt and output from the block encrypt are stored in the same data structure. ECBDATAPTR should point to this data structure before STARTECB is initiated.

| Property | Address offset | Description |
|---|---|---|
| KEY | 0 | 16 byte AES key |
| CLEARTEXT | 16 | 16 byte AES cleartext input block |
| CIPHERTEXT | 32 | 16 byte AES ciphertext output block |

*Table 32: ECB data structure overview*

## 6.6.4 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x4000E000 | ECB | ECB | AES Electronic Codebook (ECB) mode block encryption | |

*Table 33: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_STARTECB | 0x000 | Start ECB block encrypt |
| TASKS_STOPECB | 0x004 | Abort a possible executing ECB operation |
| EVENTS_ENDECB | 0x100 | ECB block encrypt complete |
| EVENTS_ERRORECB | 0x104 | ECB block encrypt aborted because of a STOPECB task or due to an error |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| ECBDATAPTR | 0x504 | ECB block encrypt memory pointers |

*Table 34: Register overview*

### 6.6.4.1 TASKS_STARTECB

Address offset: 0x000

Start ECB block encrypt

If a crypto operation is already running in the AES core, the STARTECB task will not start a new encryption and an ERRORECB event will be triggered

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STARTECB | | | Start ECB block encrypt<br>If a crypto operation is already running in the AES core, the STARTECB task will not start a new encryption and an ERRORECB event will be triggered |
| | | | Trigger | 1 | Trigger task |

### 6.6.4.2 TASKS_STOPECB

Address offset: 0x004

Abort a possible executing ECB operation

If a running ECB operation is aborted by STOPECB, the ERRORECB event is triggered.

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STOPECB | | | Abort a possible executing ECB operation<br>If a running ECB operation is aborted by STOPECB, the ERRORECB event is triggered. |
| | | | Trigger | 1 | Trigger task |

### 6.6.4.3 EVENTS_ENDECB

Address offset: 0x100

ECB block encrypt complete

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | EVENTS_ENDECB | | | ECB block encrypt complete |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.6.4.4 EVENTS_ERRORECB

Address offset: 0x104

ECB block encrypt aborted because of a STOPECB task or due to an error

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | EVENTS_ERRORECB | | | ECB block encrypt aborted because of a STOPECB task or due to an error |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.6.4.5 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | B A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | ENDECB | | | Write '1' to enable interrupt for event ENDECB |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | ERRORECB | | | Write '1' to enable interrupt for event ERRORECB |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

### 6.6.4.6 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ENDECB | | | Write '1' to disable interrupt for event ENDECB |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | ERRORECB | | | Write '1' to disable interrupt for event ERRORECB |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

### 6.6.4.7 ECBDATAPTR

Address offset: 0x504

ECB block encrypt memory pointers

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ECBDATAPTR | | | Pointer to the ECB data structure (see Table 1 ECB data structure overview) |

## 6.6.5 Electrical specification

### 6.6.5.1 ECB Electrical Specification

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{ECB}$ | Run time per 16 byte block in all modes | | 6 | | µs |

# 6.7 EGU — Event generator unit

The Event generator unit (EGU) provides support for inter-layer signaling. This means support for atomic triggering of both CPU execution and hardware tasks from both firmware (by CPU) and hardware (by PPI). This feature can, for instance, be used for triggering CPU execution at a lower priority execution from a higher priority execution, or to handle a peripheral's ISR execution at a lower priority for some of its events. However, triggering any priority from any priority is possible.

Listed here are the main EGU features:

- Enables SW triggering of interrupts
- Separate interrupt vectors for every EGU instance
- Up to 16 separate event flags per interrupt for multiplexing

Each instance of The EGU implements a set of tasks which can individually be triggered to generate the corresponding event, i.e., the corresponding event for TASKS_TRIGGER[n] is EVENTS_TRIGGERED[n].

Refer to Instances on page 113 for a list of the various EGU instances

## 6.7.1 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40014000 | EGU | EGU0 | Event generator unit 0 | |
| 0x40015000 | EGU | EGU1 | Event generator unit 1 | |

*Table 35: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_TRIGGER[0] | 0x000 | Trigger 0 for triggering the corresponding TRIGGERED[0] event |
| TASKS_TRIGGER[1] | 0x004 | Trigger 1 for triggering the corresponding TRIGGERED[1] event |
| TASKS_TRIGGER[2] | 0x008 | Trigger 2 for triggering the corresponding TRIGGERED[2] event |
| TASKS_TRIGGER[3] | 0x00C | Trigger 3 for triggering the corresponding TRIGGERED[3] event |
| TASKS_TRIGGER[4] | 0x010 | Trigger 4 for triggering the corresponding TRIGGERED[4] event |
| TASKS_TRIGGER[5] | 0x014 | Trigger 5 for triggering the corresponding TRIGGERED[5] event |
| TASKS_TRIGGER[6] | 0x018 | Trigger 6 for triggering the corresponding TRIGGERED[6] event |
| TASKS_TRIGGER[7] | 0x01C | Trigger 7 for triggering the corresponding TRIGGERED[7] event |
| TASKS_TRIGGER[8] | 0x020 | Trigger 8 for triggering the corresponding TRIGGERED[8] event |
| TASKS_TRIGGER[9] | 0x024 | Trigger 9 for triggering the corresponding TRIGGERED[9] event |
| TASKS_TRIGGER[10] | 0x028 | Trigger 10 for triggering the corresponding TRIGGERED[10] event |
| TASKS_TRIGGER[11] | 0x02C | Trigger 11 for triggering the corresponding TRIGGERED[11] event |
| TASKS_TRIGGER[12] | 0x030 | Trigger 12 for triggering the corresponding TRIGGERED[12] event |
| TASKS_TRIGGER[13] | 0x034 | Trigger 13 for triggering the corresponding TRIGGERED[13] event |
| TASKS_TRIGGER[14] | 0x038 | Trigger 14 for triggering the corresponding TRIGGERED[14] event |
| TASKS_TRIGGER[15] | 0x03C | Trigger 15 for triggering the corresponding TRIGGERED[15] event |
| EVENTS_TRIGGERED[0] | 0x100 | Event number 0 generated by triggering the corresponding TRIGGER[0] task |
| EVENTS_TRIGGERED[1] | 0x104 | Event number 1 generated by triggering the corresponding TRIGGER[1] task |
| EVENTS_TRIGGERED[2] | 0x108 | Event number 2 generated by triggering the corresponding TRIGGER[2] task |
| EVENTS_TRIGGERED[3] | 0x10C | Event number 3 generated by triggering the corresponding TRIGGER[3] task |
| EVENTS_TRIGGERED[4] | 0x110 | Event number 4 generated by triggering the corresponding TRIGGER[4] task |
| EVENTS_TRIGGERED[5] | 0x114 | Event number 5 generated by triggering the corresponding TRIGGER[5] task |
| EVENTS_TRIGGERED[6] | 0x118 | Event number 6 generated by triggering the corresponding TRIGGER[6] task |
| EVENTS_TRIGGERED[7] | 0x11C | Event number 7 generated by triggering the corresponding TRIGGER[7] task |
| EVENTS_TRIGGERED[8] | 0x120 | Event number 8 generated by triggering the corresponding TRIGGER[8] task |
| EVENTS_TRIGGERED[9] | 0x124 | Event number 9 generated by triggering the corresponding TRIGGER[9] task |
| EVENTS_TRIGGERED[10] | 0x128 | Event number 10 generated by triggering the corresponding TRIGGER[10] task |
| EVENTS_TRIGGERED[11] | 0x12C | Event number 11 generated by triggering the corresponding TRIGGER[11] task |
| EVENTS_TRIGGERED[12] | 0x130 | Event number 12 generated by triggering the corresponding TRIGGER[12] task |
| EVENTS_TRIGGERED[13] | 0x134 | Event number 13 generated by triggering the corresponding TRIGGER[13] task |
| EVENTS_TRIGGERED[14] | 0x138 | Event number 14 generated by triggering the corresponding TRIGGER[14] task |
| EVENTS_TRIGGERED[15] | 0x13C | Event number 15 generated by triggering the corresponding TRIGGER[15] task |
| INTEN | 0x300 | Enable or disable interrupt |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |

*Table 36: Register overview*

### 6.7.1.1 TASKS_TRIGGER[n] (n=0..15)

Address offset: 0x000 + (n × 0x4)

Trigger n for triggering the corresponding TRIGGERED[n] event

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_TRIGGER | | | Trigger n for triggering the corresponding TRIGGERED[n] event |
| | | | Trigger | 1 | Trigger task |

### 6.7.1.2 EVENTS_TRIGGERED[n] (n=0..15)

Address offset: 0x100 + (n × 0x4)

Event number n generated by triggering the corresponding TRIGGER[n] task

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_TRIGGERED | | | Event number n generated by triggering the corresponding TRIGGER[n] task |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.7.1.3 INTEN

Address offset: 0x300

Enable or disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-P | RW | TRIGGERED[i] (i=0..15) | | | Enable or disable interrupt for event TRIGGERED[i] |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |

### 6.7.1.4 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-P | RW | TRIGGERED[i] (i=0..15) | | | Write '1' to enable interrupt for event TRIGGERED[i] |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

### 6.7.1.5 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-P | RW | TRIGGERED[i] (i=0..15) | | | Write '1' to disable interrupt for event TRIGGERED[i] |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.7.2 Electrical specification

### 6.7.2.1 EGU Electrical Specification

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{EGU,EVT}$ | Latency between setting an EGU event flag and the system setting an interrupt | | 1 | | cycles |

# 6.8 GPIO — General purpose input/output

The general purpose input/output pins (GPIOs) are grouped as one or more ports with each port having up to 32 GPIOs.

The number of ports and GPIOs per port might vary with product variant and package. Refer to Registers on page 117 and Pin assignments on page 416 for more information about the number of GPIOs that are supported.

GPIO has the following user-configurable features:

- Up to 32 GPIO pins per GPIO port
- Configurable output drive strength
- Internal pull-up and pull-down resistors
- Wake-up from high or low level triggers on all pins
- Trigger interrupt on state changes on any pin
- All pins can be used by the PPI task/event system
- One or more GPIO outputs can be controlled through PPI and GPIOTE channels
- All pins can be individually mapped to interface blocks for layout flexibility
- GPIO state changes captured on SENSE signal can be stored by LATCH register

The GPIO port peripheral implements up to 32 pins, `PIN0` through `PIN31`. Each of these pins can be individually configured in the PIN_CNF[n] registers (n=0..31).

The following parameters can be configured through these registers:

- Direction
- Drive strength
- Enabling of pull-up and pull-down resistors
- Pin sensing
- Input buffer disconnect
- Analog input (for selected pins)

The PIN_CNF registers are retained registers. See POWER — Power supply on page 48 chapter for more information about retained registers.

## 6.8.1 Pin configuration

Pins can be individually configured, through the SENSE field in the PIN_CNF[n] register, to detect either a high level or a low level on their input.

When the correct level is detected on any such configured pin, the sense mechanism will set the DETECT signal high. Each pin has a separate DETECT signal. Default behavior, defined by the DETECTMODE register, is that the DETECT signals from all pins in the GPIO port are combined into one common DETECT signal that is routed throughout the system, which then can be utilized by other peripherals. This mechanism is functional in both System ON mode and System OFF mode. See GPIO port and the GPIO pin details on page 116.

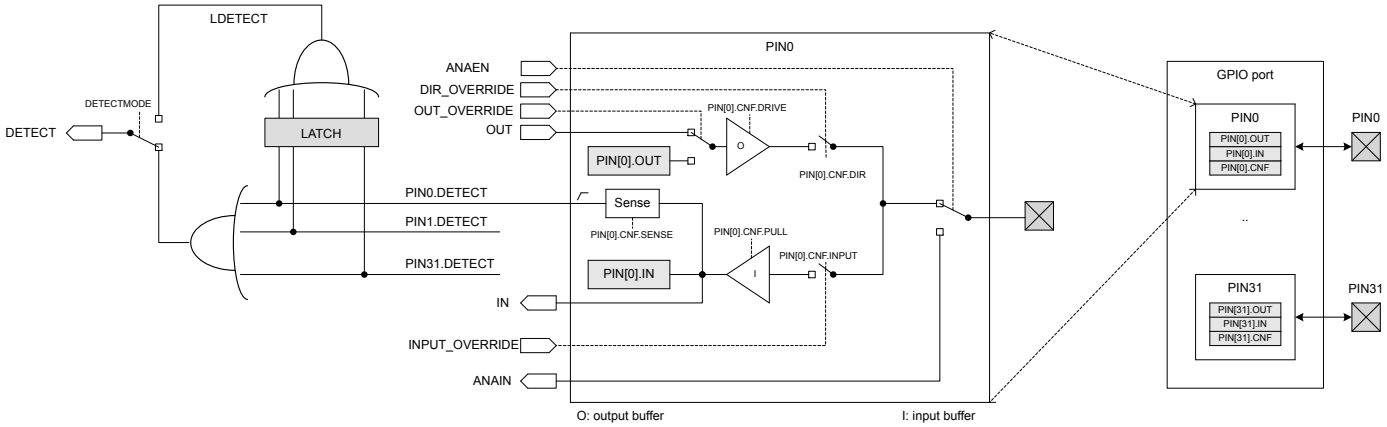



*Figure 39: GPIO port and the GPIO pin details*

GPIO port and the GPIO pin details on page 116 illustrates the GPIO port containing 32 individual pins, where `PIN0` is illustrated in more detail as a reference. All signals on the left side in the illustration are used by other peripherals in the system and therefore not directly available to the CPU.

Make sure that a pin is in a level that cannot trigger the sense mechanism before enabling it. The DETECT signal will go high immediately if the SENSE condition configured in the PIN_CNF registers is met when the sense mechanism is enabled. This will trigger a PORT event if the DETECT signal was low before enabling the sense mechanism. See GPIOTE — GPIO tasks and events on page 123.

See the following peripherals for more information about how the DETECT signal is used:

- POWER: uses the DETECT signal to exit from System OFF mode.
- GPIOTE: uses the DETECT signal to generate the PORT event.

When a pin's PINx.DETECT signal goes high, a flag will be set in the LATCH register. For example, when the PIN0.DETECT signal goes high, bit 0 in the LATCH register will be set to '1'. If the CPU performs a clear operation on a bit in the LATCH register when the associated PINx.DETECT signal is high, the bit in the LATCH register will not be cleared. The LATCH register will only be cleared if the CPU explicitly clears it by writing a '1' to the bit that shall be cleared, i.e. the LATCH register will not be affected by a PINx.DETECT signal being set low.

The LDETECT signal will be set high when one or more bits in the LATCH register are '1'. The LDETECT signal will be set low when all bits in the LATCH register are successfully cleared to '0'.

If one or more bits in the LATCH register are '1' after the CPU has performed a clear operation on the LATCH registers, a rising edge will be generated on the LDETECT signal. This is illustrated in DETECT signal behavior on page 117.

**Important:** The CPU can read the LATCH register at any time to check if a SENSE condition has been met on one or more of the the GPIO pins, even if that condition is no longer met at the time the CPU queries the LATCH register. This mechanism will work even if the LDETECT signal is not used as the DETECT signal.

The LDETECT signal is by default not connected to the GPIO port's DETECT signal, but via the DETECTMODE register it is possible to change from default behavior to DETECT signal being derived directly from the LDETECT signal instead. See GPIO port and the GPIO pin details on page 116. DETECT signal behavior on page 117 illustrates the DETECT signal behavior for these two alternatives.

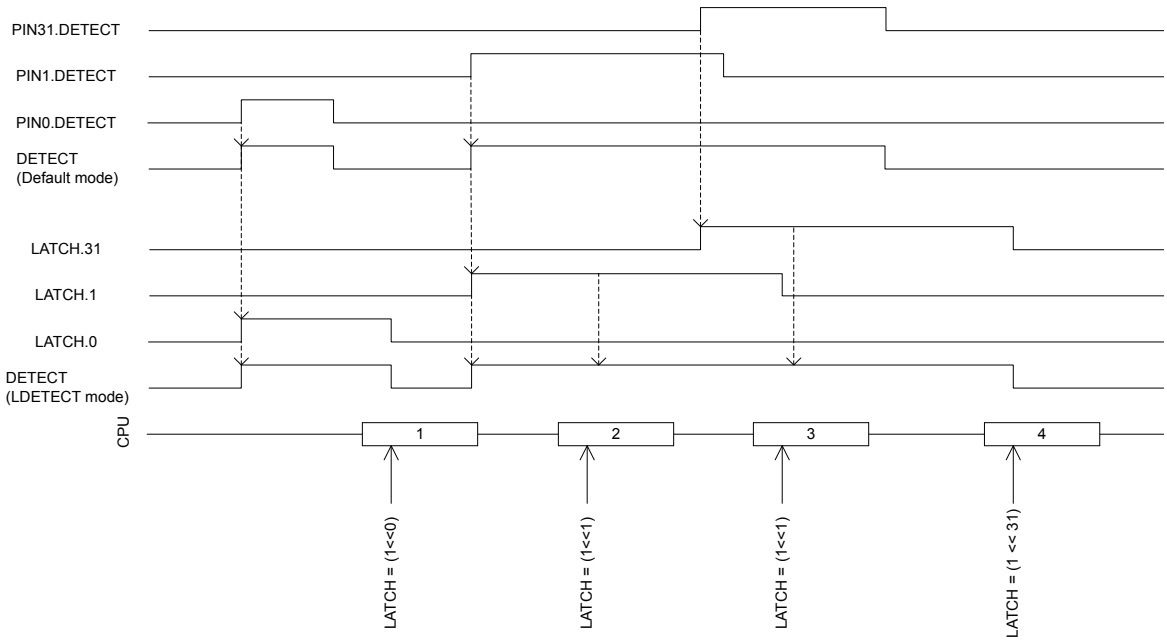



*Figure 40: DETECT signal behavior*

The input buffer of a GPIO pin can be disconnected from the pin to enable power savings when the pin is not used as an input, see GPIO port and the GPIO pin details on page 116. Inputs must be connected to get a valid input value in the IN register, and for the sense mechanism to get access to the pin.

Other peripherals in the system can connect to GPIO pins and override their output value and configuration, or read their analog or digital input value. See GPIO port and the GPIO pin details on page 116.

Selected pins also support analog input signals, see ANAIN in GPIO port and the GPIO pin details on page 116. The assignment of the analog pins can be found in Pin assignments on page 416.

**Important:** When a pin is configured as digital input, care has been taken to minimize increased current consumption when the input voltage is between $V_{IL}$ and $V_{IH}$. However, it is a good practice to ensure that the external circuitry does not drive that pin to levels between $V_{IL}$ and $V_{IH}$ for a long period of time.

## 6.8.2 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x50000000 | GPIO | P0 | General purpose input and output | |

*Table 37: Instances*

| Register | Offset | Description |
|---|---|---|
| OUT | 0x504 | Write GPIO port |

| Register | Offset | Description |
|---|---|---|
| OUTSET | 0x508 | Set individual bits in GPIO port |
| OUTCLR | 0x50C | Clear individual bits in GPIO port |
| IN | 0x510 | Read GPIO port |
| DIR | 0x514 | Direction of GPIO pins |
| DIRSET | 0x518 | DIR set register |
| DIRCLR | 0x51C | DIR clear register |
| LATCH | 0x520 | Latch register indicating what GPIO pins that have met the criteria set in the PIN_CNF[n].SENSE registers |
| DETECTMODE | 0x524 | Select between default DETECT signal behaviour and LDETECT mode |
| PIN_CNF[0] | 0x700 | Configuration of GPIO pins |
| PIN_CNF[1] | 0x704 | Configuration of GPIO pins |
| PIN_CNF[2] | 0x708 | Configuration of GPIO pins |
| PIN_CNF[3] | 0x70C | Configuration of GPIO pins |
| PIN_CNF[4] | 0x710 | Configuration of GPIO pins |
| PIN_CNF[5] | 0x714 | Configuration of GPIO pins |
| PIN_CNF[6] | 0x718 | Configuration of GPIO pins |
| PIN_CNF[7] | 0x71C | Configuration of GPIO pins |
| PIN_CNF[8] | 0x720 | Configuration of GPIO pins |
| PIN_CNF[9] | 0x724 | Configuration of GPIO pins |
| PIN_CNF[10] | 0x728 | Configuration of GPIO pins |
| PIN_CNF[11] | 0x72C | Configuration of GPIO pins |
| PIN_CNF[12] | 0x730 | Configuration of GPIO pins |
| PIN_CNF[13] | 0x734 | Configuration of GPIO pins |
| PIN_CNF[14] | 0x738 | Configuration of GPIO pins |
| PIN_CNF[15] | 0x73C | Configuration of GPIO pins |
| PIN_CNF[16] | 0x740 | Configuration of GPIO pins |
| PIN_CNF[17] | 0x744 | Configuration of GPIO pins |
| PIN_CNF[18] | 0x748 | Configuration of GPIO pins |
| PIN_CNF[19] | 0x74C | Configuration of GPIO pins |
| PIN_CNF[20] | 0x750 | Configuration of GPIO pins |
| PIN_CNF[21] | 0x754 | Configuration of GPIO pins |
| PIN_CNF[22] | 0x758 | Configuration of GPIO pins |
| PIN_CNF[23] | 0x75C | Configuration of GPIO pins |
| PIN_CNF[24] | 0x760 | Configuration of GPIO pins |
| PIN_CNF[25] | 0x764 | Configuration of GPIO pins |
| PIN_CNF[26] | 0x768 | Configuration of GPIO pins |
| PIN_CNF[27] | 0x76C | Configuration of GPIO pins |
| PIN_CNF[28] | 0x770 | Configuration of GPIO pins |
| PIN_CNF[29] | 0x774 | Configuration of GPIO pins |
| PIN_CNF[30] | 0x778 | Configuration of GPIO pins |
| PIN_CNF[31] | 0x77C | Configuration of GPIO pins |

*Table 38: Register overview*

### 6.8.2.1 OUT

Address offset: 0x504

Write GPIO port

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | f e d c b a Z Y X W V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce Field | Value ID | Value Description |
| A-f | RW PIN[i] (i=0..31) | | Pin i |
| | | Low | 0 Pin driver is low |
| | | High | 1 Pin driver is high |

### 6.8.2.2 OUTSET

Address offset: 0x508

Set individual bits in GPIO port

Read: reads value of OUT register.

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | f e d c b a Z Y X W V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce Field | Value ID | Value Description |
| A-f | RW PIN[i] (i=0..31) | | Pin i |
| | | Low | 0 Read: pin driver is low |
| | | High | 1 Read: pin driver is high |
| | | Set | 1 Write: writing a '1' sets the pin high; writing a '0' has no effect |

### 6.8.2.3 OUTCLR

Address offset: 0x50C

Clear individual bits in GPIO port

Read: reads value of OUT register.

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | f e d c b a Z Y X W V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce Field | Value ID | Value Description |
| A-f | RW PIN[i] (i=0..31) | | Pin i |
| | | Low | 0 Read: pin driver is low |
| | | High | 1 Read: pin driver is high |
| | | Clear | 1 Write: writing a '1' sets the pin low; writing a '0' has no effect |

### 6.8.2.4 IN

Address offset: 0x510

Read GPIO port

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | f e d c b a Z Y X W V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-f | R | PIN[i] (i=0..31) | | | Pin i |
| | | | Low | 0 | Pin input is low |
| | | | High | 1 | Pin input is high |

### 6.8.2.5 DIR

Address offset: 0x514

Direction of GPIO pins

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | f e d c b a Z Y X W V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-f | RW | PIN[i] (i=0..31) | | | Pin i |
| | | | Input | 0 | Pin set as input |
| | | | Output | 1 | Pin set as output |

### 6.8.2.6 DIRSET

Address offset: 0x518

DIR set register

Read: reads value of DIR register.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | f e d c b a Z Y X W V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-f | RW | PIN[i] (i=0..31) | | | Set as output pin i |
| | | | Input | 0 | Read: pin set as input |
| | | | Output | 1 | Read: pin set as output |
| | | | Set | 1 | Write: writing a '1' sets pin to output; writing a '0' has no effect |

### 6.8.2.7 DIRCLR

Address offset: 0x51C

DIR clear register

Read: reads value of DIR register.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | f e d c b a Z Y X W V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-f | RW | PIN[i] (i=0..31) | | | Set as input pin i |
| | | | Input | 0 | Read: pin set as input |
| | | | Output | 1 | Read: pin set as output |
| | | | Clear | 1 | Write: writing a '1' sets pin to input; writing a '0' has no effect |

### 6.8.2.8 LATCH

Address offset: 0x520

Latch register indicating what GPIO pins that have met the criteria set in the PIN_CNF[n].SENSE registers

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | f e d c b a Z Y X W V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-f | RW | PIN[i] (i=0..31) | | | Status on whether PINi has met criteria set in PIN_CNFi.SENSE register. Write '1' to clear. |
| | | | NotLatched | 0 | Criteria has not been met |
| | | | Latched | 1 | Criteria has been met |

### 6.8.2.9 DETECTMODE

Address offset: 0x524

Select between default DETECT signal behaviour and LDETECT mode

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | DETECTMODE | | | Select between default DETECT signal behaviour and LDETECT mode |
| | | | Default | 0 | DETECT directly connected to PIN DETECT signals |
| | | | LDETECT | 1 | Use the latched LDETECT behaviour |

### 6.8.2.10 PIN_CNF[n] (n=0..31)

Address offset: 0x700 + (n × 0x4)

Configuration of GPIO pins

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | E E D D D C C B A |
| **Reset 0x00000002** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | DIR | | | Pin direction. Same physical register as DIR register |
| | | | Input | 0 | Configure pin as an input pin |
| | | | Output | 1 | Configure pin as an output pin |
| B | RW | INPUT | | | Connect or disconnect input buffer |
| | | | Connect | 0 | Connect input buffer |

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | E E D D D C C B A |
| **Reset 0x00000002** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| | | | Disconnect | 1 | Disconnect input buffer |
| C | RW | PULL | | | Pull configuration |
| | | | Disabled | 0 | No pull |
| | | | Pulldown | 1 | Pull down on pin |
| | | | Pullup | 3 | Pull up on pin |
| D | RW | DRIVE | | | Drive configuration |
| | | | S0S1 | 0 | Standard '0', standard '1' |
| | | | H0S1 | 1 | High drive '0', standard '1' |
| | | | S0H1 | 2 | Standard '0', high drive '1' |
| | | | H0H1 | 3 | High drive '0', high 'drive '1'' |
| | | | D0S1 | 4 | Disconnect '0' standard '1' (normally used for wired-or connections) |
| | | | D0H1 | 5 | Disconnect '0', high drive '1' (normally used for wired-or connections) |
| | | | S0D1 | 6 | Standard '0'. disconnect '1' (normally used for wired-and connections) |
| | | | H0D1 | 7 | High drive '0', disconnect '1' (normally used for wired-and connections) |
| E | RW | SENSE | | | Pin sensing mechanism |
| | | | Disabled | 0 | Disabled |
| | | | High | 2 | Sense for high level |
| | | | Low | 3 | Sense for low level |

## 6.8.3 Electrical specification

### 6.8.3.1 GPIO Electrical Specification

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $V_{IH}$ | Input high voltage | 0.7 x VDD | | VDD | V |
| $V_{IL}$ | Input low voltage | VSS | | 0.3 x VDD | V |
| $V_{OH,SD}$ | Output high voltage, standard drive, 0.5 mA, VDD ≥1.7 | VDD-0.4 | | VDD | V |
| $V_{OH,HDH}$ | Output high voltage, high drive, 5 mA, VDD >= 2.7 V | VDD-0.4 | | VDD | V |
| $V_{OH,HDL}$ | Output high voltage, high drive, 3 mA, VDD >= 1.7 V | VDD-0.4 | | VDD | V |
| $V_{OL,SD}$ | Output low voltage, standard drive, 0.5 mA, VDD ≥1.7 | VSS | | VSS+0.4 | V |
| $V_{OL,HDH}$ | Output low voltage, high drive, 5 mA, VDD >= 2.7 V | VSS | | VSS+0.4 | V |
| $V_{OL,HDL}$ | Output low voltage, high drive, 3 mA, VDD >= 1.7 V | VSS | | VSS+0.4 | V |
| $I_{OL,SD}$ | Current at VSS+0.4 V, output set low, standard drive, VDD ≥1.7 | 1 | 2 | 4 | mA |
| $I_{OL,HDH}$ | Current at VSS+0.4 V, output set low, high drive, VDD >= 2.7 V | 6 | 10 | 15 | mA |
| $I_{OL,HDL}$ | Current at VSS+0.4 V, output set low, high drive, VDD >= 1.7 V | 3 | | | mA |
| $I_{OH,SD}$ | Current at VDD-0.4 V, output set high, standard drive, VDD ≥1.7 | 1 | 2 | 4 | mA |
| $I_{OH,HDH}$ | Current at VDD-0.4 V, output set high, high drive, VDD >= 2.7 V | 6 | 9 | 14 | mA |

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $I_{OH,HDL}$ | Current at VDD-0.4 V, output set high, high drive, VDD >= 1.7 V | 3 | | | mA |
| $t_{RF,15pF}$ | Rise/fall time, standard drive mode, 10-90%, 15 pF load[1] | | 9 | | ns |
| $t_{RF,25pF}$ | Rise/fall time, standard drive mode, 10-90%, 25 pF load[1] | | 13 | | ns |
| $t_{RF,50pF}$ | Rise/fall time, standard drive mode, 10-90%, 50 pF load[1] | | 25 | | ns |
| $t_{HRF,15pF}$ | Rise/Fall time, high drive mode, 10-90%, 15 pF load[1] | | 4 | | ns |
| $t_{HRF,25pF}$ | Rise/Fall time, high drive mode, 10-90%, 25 pF load[1] | | 5 | | ns |
| $t_{HRF,50pF}$ | Rise/Fall time, high drive mode, 10-90%, 50 pF load[1] | | 8 | | ns |
| $R_{PU}$ | Pull-up resistance | 11 | 13 | 16 | kΩ |
| $R_{PD}$ | Pull-down resistance | 11 | 13 | 16 | kΩ |
| $C_{PAD}$ | Pad capacitance | | 3 | | pF |

# 6.9 GPIOTE — GPIO tasks and events

The GPIO tasks and events (GPIOTE) module provides functionality for accessing GPIO pins using tasks and events. Each GPIOTE channel can be assigned to one pin.

A GPIOTE block enables GPIOs to generate events on pin state change which can be used to carry out tasks through the PPI system. A GPIO can also be driven to change state on system events using the PPI system. Low power detection of pin state changes is possible when in System ON or System OFF.

| Instance | Number of GPIOTE channels |
|---|---|
| GPIOTE | 8 |

*Table 39: GPIOTE properties*

Up to three tasks can be used in each GPIOTE channel for performing write operations to a pin. Two tasks are fixed (SET and CLR), and one (OUT) is configurable to perform following operations:

- Set
- Clear
- Toggle

An event can be generated in each GPIOTE channel from one of the following input conditions:

- Rising edge
- Falling edge
- Any change

## 6.9.1 Pin events and tasks

The GPIOTE module has a number of tasks and events that can be configured to operate on individual GPIO pins.

The tasks (SET[n], CLR[n] and OUT[n]) can be used for writing to individual pins, and the events (IN[n]) can be generated from changes occurring at the inputs of individual pins.

The SET task will set the pin selected in CONFIG[n].PSEL to high.

The CLR task will set the pin low.

The effect of the OUT task on the pin is configurable in CONFIG[n].POLARITY , and can either set the pin high, set it low, or toggle it.

[1] Rise and fall times based on simulations

The tasks and events are configured using the CONFIG[n] registers. Every set of SET, CLR and OUT[n] tasks and IN[n] events has one CONFIG[n] register associated with it.

As long as a SET[n], CLR[n] and OUT[n] task or an IN[n] event is configured to control a pin **n**, the pin's output value will only be updated by the GPIOTE module. The pin's output value as specified in the GPIO will therefore be ignored as long as the pin is controlled by GPIOTE. Attempting to write a pin as a normal GPIO pin will have no effect. When the GPIOTE is disconnected from a pin, see MODE field in CONFIG[n] register, the associated pin will get the output and configuration values specified in the GPIO module.

When conflicting tasks are triggered simultaneously (i.e. during the same clock cycle) in one channel, the precedence of the tasks will be as described in Task priorities on page 124.

| Priority | Task |
|---|---|
| 1 | OUT |
| 2 | CLR |
| 3 | SET |

*Table 40: Task priorities*

When setting the CONFIG[n] registers, MODE=Disabled does not have the same effect as MODE=Task and POLARITY=None. In the latter case, a CLR or SET task occurring at the exact same time as OUT will end up with no change on the pin, according to the priorities described in the table above.

When a GPIOTE channel is configured to operate on a pin as a task, the initial value of that pin is configured in the OUTINIT field of CONFIG[n].

## 6.9.2 Port event

PORT is an event that can be generated from multiple input pins using the GPIO DETECT signal.

The event will be generated on the rising edge of the DETECT signal. See GPIO — General purpose input/output on page 115 for more information about the DETECT signal.

Putting the system into System ON IDLE while DETECT is high will not cause DETECT to wake the system up again. Make sure to clear all DETECT sources before entering sleep. If the LATCH register is used as a source, if any bit in LATCH is still high after clearing all or part of the register (for instance due to one of the PINx.DETECT signal still high), a new rising edge will be generated on DETECT, see Pin configuration on page 116.

Trying to put the system to System OFF while DETECT is high will cause a wakeup from System OFF reset.

This feature is always enabled although the peripheral itself appears to be IDLE, that is, no clocks or other power intensive infrastructure have to be requested to keep this feature enabled. This feature can therefore be used to wake up the CPU from a WFI or WFE type sleep in System ON with all peripherals and the CPU idle, that is, lowest power consumption in System ON mode.

In order to prevent spurious interrupts from the PORT event while configuring the sources, the user shall first disable interrupts on the PORT event (through INTENCLR.PORT), then configure the sources (PIN_CNF[n].SENSE), clear any potential event that could have occurred during configuration (write '0' to EVENTS_PORT), and finally enable interrupts (through INTENSET.PORT).

## 6.9.3 Tasks and events pin configuration

Each GPIOTE channel is associated with one physical GPIO pin through the CONFIG.PSEL field.

When Event mode is selected in CONFIG.MODE, the pin specified by CONFIG.PSEL will be configured as an input, overriding the DIR setting in GPIO. Similarly, when Task mode is selected in CONFIG.MODE, the pin specified by CONFIG.PSEL will be configured as an output overriding the DIR setting and OUT value in GPIO. When Disabled is selected in CONFIG.MODE, the pin specified by CONFIG.PSEL will use its configuration from the PIN[n].CNF registers in GPIO.

Only one GPIOTE channel can be assigned to one physical pin. Failing to do so may result in unpredictable behavior.

## 6.9.4 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40006000 | GPIOTE | GPIOTE | GPIO tasks and events | |

*Table 41: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_OUT[0] | 0x000 | Task for writing to pin specified in CONFIG[0].PSEL. Action on pin is configured in CONFIG[0].POLARITY. |
| TASKS_OUT[1] | 0x004 | Task for writing to pin specified in CONFIG[1].PSEL. Action on pin is configured in CONFIG[1].POLARITY. |
| TASKS_OUT[2] | 0x008 | Task for writing to pin specified in CONFIG[2].PSEL. Action on pin is configured in CONFIG[2].POLARITY. |
| TASKS_OUT[3] | 0x00C | Task for writing to pin specified in CONFIG[3].PSEL. Action on pin is configured in CONFIG[3].POLARITY. |
| TASKS_OUT[4] | 0x010 | Task for writing to pin specified in CONFIG[4].PSEL. Action on pin is configured in CONFIG[4].POLARITY. |
| TASKS_OUT[5] | 0x014 | Task for writing to pin specified in CONFIG[5].PSEL. Action on pin is configured in CONFIG[5].POLARITY. |
| TASKS_OUT[6] | 0x018 | Task for writing to pin specified in CONFIG[6].PSEL. Action on pin is configured in CONFIG[6].POLARITY. |
| TASKS_OUT[7] | 0x01C | Task for writing to pin specified in CONFIG[7].PSEL. Action on pin is configured in CONFIG[7].POLARITY. |
| TASKS_SET[0] | 0x030 | Task for writing to pin specified in CONFIG[0].PSEL. Action on pin is to set it high. |
| TASKS_SET[1] | 0x034 | Task for writing to pin specified in CONFIG[1].PSEL. Action on pin is to set it high. |
| TASKS_SET[2] | 0x038 | Task for writing to pin specified in CONFIG[2].PSEL. Action on pin is to set it high. |
| TASKS_SET[3] | 0x03C | Task for writing to pin specified in CONFIG[3].PSEL. Action on pin is to set it high. |
| TASKS_SET[4] | 0x040 | Task for writing to pin specified in CONFIG[4].PSEL. Action on pin is to set it high. |
| TASKS_SET[5] | 0x044 | Task for writing to pin specified in CONFIG[5].PSEL. Action on pin is to set it high. |
| TASKS_SET[6] | 0x048 | Task for writing to pin specified in CONFIG[6].PSEL. Action on pin is to set it high. |
| TASKS_SET[7] | 0x04C | Task for writing to pin specified in CONFIG[7].PSEL. Action on pin is to set it high. |
| TASKS_CLR[0] | 0x060 | Task for writing to pin specified in CONFIG[0].PSEL. Action on pin is to set it low. |
| TASKS_CLR[1] | 0x064 | Task for writing to pin specified in CONFIG[1].PSEL. Action on pin is to set it low. |
| TASKS_CLR[2] | 0x068 | Task for writing to pin specified in CONFIG[2].PSEL. Action on pin is to set it low. |
| TASKS_CLR[3] | 0x06C | Task for writing to pin specified in CONFIG[3].PSEL. Action on pin is to set it low. |
| TASKS_CLR[4] | 0x070 | Task for writing to pin specified in CONFIG[4].PSEL. Action on pin is to set it low. |
| TASKS_CLR[5] | 0x074 | Task for writing to pin specified in CONFIG[5].PSEL. Action on pin is to set it low. |
| TASKS_CLR[6] | 0x078 | Task for writing to pin specified in CONFIG[6].PSEL. Action on pin is to set it low. |
| TASKS_CLR[7] | 0x07C | Task for writing to pin specified in CONFIG[7].PSEL. Action on pin is to set it low. |
| EVENTS_IN[0] | 0x100 | Event generated from pin specified in CONFIG[0].PSEL |
| EVENTS_IN[1] | 0x104 | Event generated from pin specified in CONFIG[1].PSEL |
| EVENTS_IN[2] | 0x108 | Event generated from pin specified in CONFIG[2].PSEL |
| EVENTS_IN[3] | 0x10C | Event generated from pin specified in CONFIG[3].PSEL |
| EVENTS_IN[4] | 0x110 | Event generated from pin specified in CONFIG[4].PSEL |
| EVENTS_IN[5] | 0x114 | Event generated from pin specified in CONFIG[5].PSEL |
| EVENTS_IN[6] | 0x118 | Event generated from pin specified in CONFIG[6].PSEL |
| EVENTS_IN[7] | 0x11C | Event generated from pin specified in CONFIG[7].PSEL |
| EVENTS_PORT | 0x17C | Event generated from multiple input GPIO pins with SENSE mechanism enabled |
| INTENSET | 0x304 | Enable interrupt |

| Register | Offset | Description |
|---|---|---|
| INTENCLR | 0x308 | Disable interrupt |
| CONFIG[0] | 0x510 | Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event |
| CONFIG[1] | 0x514 | Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event |
| CONFIG[2] | 0x518 | Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event |
| CONFIG[3] | 0x51C | Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event |
| CONFIG[4] | 0x520 | Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event |
| CONFIG[5] | 0x524 | Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event |
| CONFIG[6] | 0x528 | Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event |
| CONFIG[7] | 0x52C | Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event |

*Table 42: Register overview*

### 6.9.4.1 TASKS_OUT[n] (n=0..7)

Address offset: 0x000 + (n × 0x4)

Task for writing to pin specified in CONFIG[n].PSEL. Action on pin is configured in CONFIG[n].POLARITY.

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_OUT | | | Task for writing to pin specified in CONFIG[n].PSEL. Action on pin is configured in CONFIG[n].POLARITY. |
| | | | Trigger | 1 | Trigger task |

### 6.9.4.2 TASKS_SET[n] (n=0..7)

Address offset: 0x030 + (n × 0x4)

Task for writing to pin specified in CONFIG[n].PSEL. Action on pin is to set it high.

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_SET | | | Task for writing to pin specified in CONFIG[n].PSEL. Action on pin is to set it high. |
| | | | Trigger | 1 | Trigger task |

### 6.9.4.3 TASKS_CLR[n] (n=0..7)

Address offset: 0x060 + (n × 0x4)

Task for writing to pin specified in CONFIG[n].PSEL. Action on pin is to set it low.

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_CLR | | | Task for writing to pin specified in CONFIG[n].PSEL. Action on pin is to set it low. |
| | | | Trigger | 1 | Trigger task |

### 6.9.4.4 EVENTS_IN[n] (n=0..7)

Address offset: 0x100 + (n × 0x4)

Event generated from pin specified in CONFIG[n].PSEL

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_IN | | | Event generated from pin specified in CONFIG[n].PSEL |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.9.4.5 EVENTS_PORT

Address offset: 0x17C

Event generated from multiple input GPIO pins with SENSE mechanism enabled

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_PORT | | | Event generated from multiple input GPIO pins with SENSE mechanism enabled |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.9.4.6 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-H | RW | IN[i] (i=0..7) | | | Write '1' to enable interrupt for event IN[i] |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| I | RW | PORT | | | Write '1' to enable interrupt for event PORT |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

### 6.9.4.7 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-H | RW | IN[i] (i=0..7) | | | Write '1' to disable interrupt for event IN[i] |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| I | RW | PORT | | | Write '1' to disable interrupt for event PORT |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.9.4.8 CONFIG[n] (n=0..7)

Address offset: 0x510 + (n × 0x4)

Configuration for OUT[n], SET[n] and CLR[n] tasks and IN[n] event

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | E D D B B B B B A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | MODE | | | Mode |
| | | | Disabled | 0 | Disabled. Pin specified by PSEL will not be acquired by the GPIOTE module. |
| | | | Event | 1 | Event mode<br>The pin specified by PSEL will be configured as an input and the IN[n] event will be generated if operation specified in POLARITY occurs on the pin. |
| | | | Task | 3 | Task mode<br>The GPIO specified by PSEL will be configured as an output and triggering the SET[n], CLR[n] or OUT[n] task will perform the operation specified by POLARITY on the pin. When enabled as a task the GPIOTE module will acquire the pin and the pin can no longer be written as a regular output pin from the GPIO module. |
| B | RW | PSEL | | [0..31] | GPIO number associated with SET[n], CLR[n] and OUT[n] tasks and IN[n] event |
| D | RW | POLARITY | | | When In task mode: Operation to be performed on output when OUT[n] task is triggered. When In event mode: Operation on input that shall trigger IN[n] event. |
| | | | None | 0 | Task mode: No effect on pin from OUT[n] task. Event mode: no IN[n] event generated on pin activity. |
| | | | LoToHi | 1 | Task mode: Set pin from OUT[n] task. Event mode: Generate IN[n] event when rising edge on pin. |
| | | | HiToLo | 2 | Task mode: Clear pin from OUT[n] task. Event mode: Generate IN[n] event when falling edge on pin. |
| | | | Toggle | 3 | Task mode: Toggle pin from OUT[n]. Event mode: Generate IN[n] when any change on pin. |
| E | RW | OUTINIT | | | When in task mode: Initial value of the output when the GPIOTE channel is configured. When in event mode: No effect. |
| | | | Low | 0 | Task mode: Initial value of pin before task triggering is low |

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | E D D B B B B B A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce Field | Value ID | Value Description |
| | | High | 1 Task mode: Initial value of pin before task triggering is high |

## 6.9.5 Electrical specification

# 6.10 PDM — Pulse density modulation interface

The pulse density modulation (PDM) module enables input of pulse density modulated signals from external audio frontends, for example, digital microphones. The PDM module generates the PDM clock and supports single-channel or dual-channel (Left and Right) data input. Data is transferred directly to RAM buffers using EasyDMA.

Listed here are the main features for PDM:

- Up to two PDM microphones configured as a Left/Right pair using the same data input
- 16 kHz output sample rate, 16-bit samples
- EasyDMA support for sample buffering
- HW decimation filters

The PDM module illustrated in PDM module on page 129 is interfacing up to two digital microphones with the PDM interface. It implements EasyDMA, which relieves real-time requirements associated with controlling the PDM slave from a low priority CPU execution context. It also includes all the necessary digital filter elements to produce PCM samples. The PDM module allows continuous audio streaming.

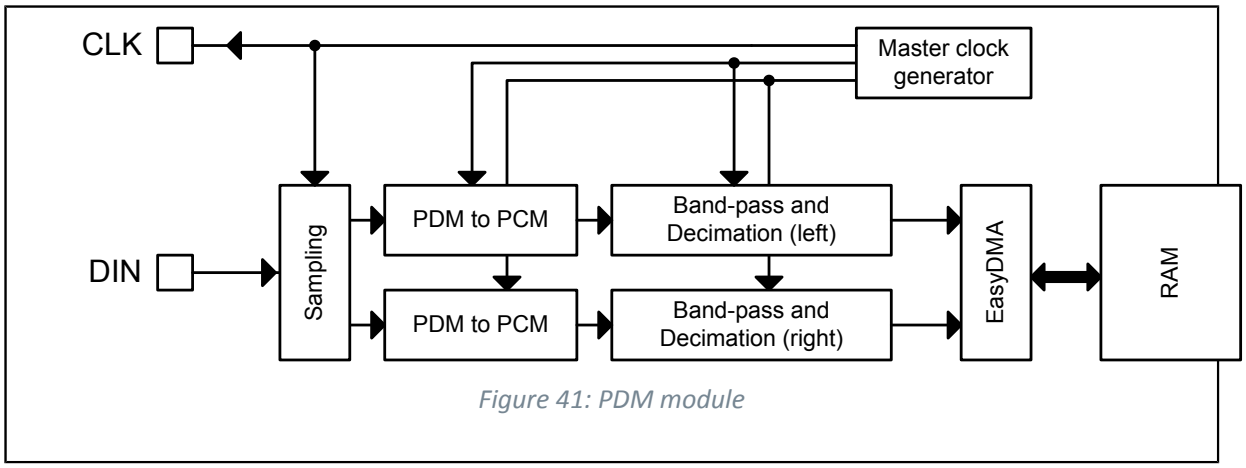



*Figure 41: PDM module*

## 6.10.1 Master clock generator

The FREQ field in the master clock's PDMCLKCTRL register allows adjusting the PDM clock's frequency.

The master clock generator does not add any jitter to the HFCLK source chosen. It is recommended (but not mandatory) to use the Xtal as HFCLK source.

## 6.10.2 Module operation

By default, bits from the left PDM microphone are sampled on PDM_CLK falling edge, bits for the right are sampled on the rising edge of PDM_CLK, resulting in two bitstreams. Each bitstream is fed into a digital filter which converts the PDM stream into 16-bit PCM samples, and filters and down-samples them to reach the appropriate sample rate.

The EDGE field in the MODE register allows swapping Left and Right, so that Left will be sampled on rising edge, and Right on falling.

The PDM module uses EasyDMA to store the samples coming out from the filters into one buffer in RAM.

Depending on the mode chosen in the OPERATION field in the MODE register, memory either contains alternating left and right 16-bit samples (Stereo), or only left 16-bit samples (Mono).

To ensure continuous PDM sampling, it is up to the application to update the EasyDMA destination address pointer as the previous buffer is filled.

The continuous transfer can be started or stopped by sending the START and STOP tasks. STOP becomes effective after the current frame has finished transferring, which will generate the STOPPED event. The STOPPED event indicates that all activity in the module are finished, and that the data is available in RAM (EasyDMA has finished transferring as well). Attempting to restart before receiving the STOPPED event may result in unpredictable behaviour.

## 6.10.3 Decimation filter

In order to convert the incoming data stream into PCM audio samples, a decimation filter is included in the PDM interface module.

The input of the filter is the two-channel PDM serial stream (with left channel on clock high, right channel on clock low), its output is 2 × 16-bit PCM samples at a sample rate 64 times lower than the PDM clock rate.

The filter stage of each channel is followed by a digital volume control, to attenuate or amplify the output samples in a range of -20 dB to +20 dB around the default (reset) setting, defined by $G_{PDM,default}$. The gain is controlled by the GAINL and GAINR registers.

As an example, if the goal is to achieve 2500 RMS output samples (16 bit) with a 1 kHz 90 dBA signal into a -26 dBFS sensitivity PDM microphone, the user will have to sum the PDM module's default gain ( $G_{PDM,default}$ ) and the gain introduced by the microphone and acoustic path of his implementation (an attenuation would translate into a negative gain), and adjust GAINL and GAINR by this amount. Assuming that only the PDM module influences the gain, GAINL and GAINR must be set to $-G_{PDM,default}$ dB to achieve the requirement.

With $G_{PDM,default}$=3.2 dB, and as GAINL and GAINR are expressed in 0.5 dB steps, the closest value to program would be 3.0 dB, which can be calculated as:

```
GAINL = GAINR = (DefaultGain - (2 * 3))
```

Remember to check that the resulting values programmed into GAINL and GAINR fall within MinGain and MaxGain.

## 6.10.4 EasyDMA

Samples will be written directly to RAM, and EasyDMA must be configured accordingly.

The address pointer for the EasyDMA channel is set in SAMPLE.PTR register. If the destination address set in SAMPLE.PTR is not pointing to the Data RAM region, an EasyDMA transfer may result in a HardFault or RAM corruption. See Memory on page 16 for more information about the different memory regions.

DMA supports Stereo (Left+Right 16-bit samples) and Mono (Left only) data transfer, depending on setting in the OPERATION field in the MODE register. The samples are stored little endian.

| MODE.OPERATION | Bits per sample | Result stored per RAM word | Physical RAM allocated (32 bit words) | Result boundary indexes in RAM | Note |
|---|---|---|---|---|---|
| Stereo | 32 (2x16) | L+R | ceil(SAMPLE.MAXCNT/2) | R0=[31:16]; L0=[15:0] | Default |
| Mono | 16 | 2xL | ceil(SAMPLE.MAXCNT/2) | L1=[31:16]; L0=[15:0] | |

*Table 43: DMA sample storage*

The destination buffer in RAM consists of one block, the size of which is set in SAMPLE.MAXCNT register. Format is number of 16-bit samples. The physical RAM allocated is always:

```
(RAM allocation, in bytes) = SAMPLE.MAXCNT * 2;
```

(but the mapping of the samples depends on MODE.OPERATION.

If OPERATION=Stereo, RAM will contain a succession of Left and Right samples.

If OPERATION=Mono, RAM will contain a succession of mono samples.

For a given value of SAMPLE.MAXCNT, the buffer in RAM can contain half the stereo sampling time as compared to the mono sampling time.

The PDM acquisition can be started by the START task, after the SAMPLE.PTR and SAMPLE.MAXCNT registers have been written. When starting the module, it will take some time for the filters to start outputting valid data. Transients from the PDM microphone itself may also occur. The first few samples (typically around 50) might hence contain invalid values or transients. It is therefore advised to discard the first few samples after a PDM start.

As soon as the STARTED event is received, the firmware can write the next SAMPLE.PTR value (this register is double-buffered), to ensure continuous operation.

When the buffer in RAM is filled with samples, an END event is triggered. The firmware can start processing the data in the buffer. Meanwhile, the PDM module starts acquiring data into the new buffer pointed to by SAMPLE.PTR, and sends a new STARTED event, so that the firmware can update SAMPLE.PTR to the next buffer address.

## 6.10.5 Hardware example

Connect the microphone clock to CLK, and data to DIN.

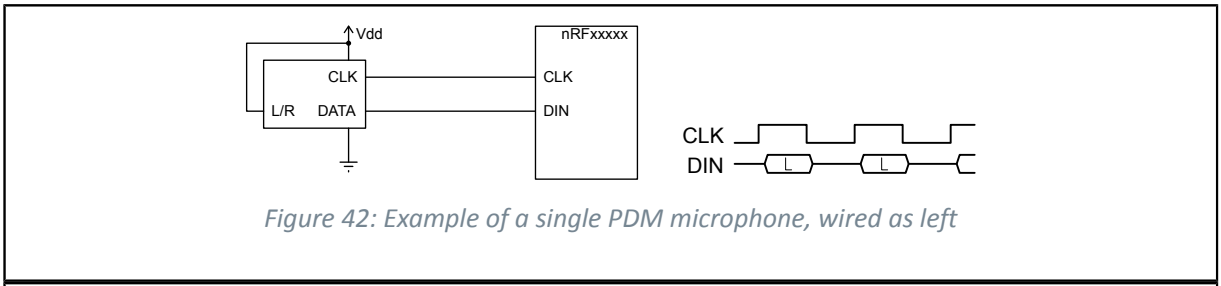



*Figure 42: Example of a single PDM microphone, wired as left*

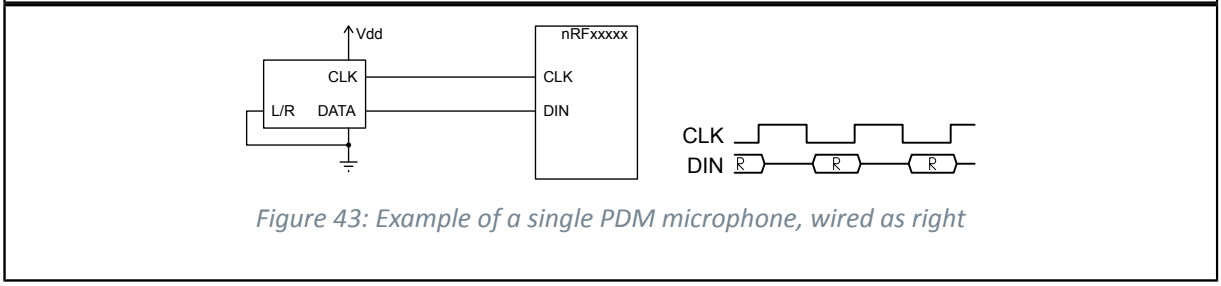



*Figure 43: Example of a single PDM microphone, wired as right*

Note that in a single-microphone (mono) configuration, depending on the microphone's implementation, either the left or the right channel (sampled at falling or rising CLK edge respectively) will contain reliable data. If two microphones are used, one of them has to be set as left, the other as right (L/R pin tied high or to GND on the respective microphone). It is strongly recommended to use two microphones of exactly the same brand and type so that their timings in left and right operation match.

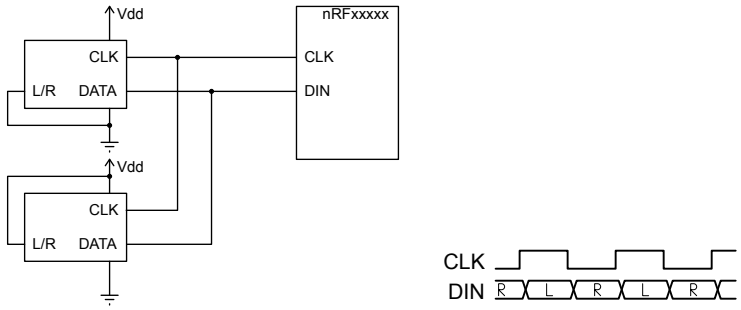



*Figure 44: Example of two PDM microphones*

## 6.10.6 Pin configuration

The CLK and DIN signals associated to the PDM module are mapped to physical pins according to the configuration specified in the PSEL.CLK and PSEL.DIN registers respectively. If the CONNECT field in any PSEL register is set to Disconnected, the associated PDM module signal will not be connected to the required physical pins, and will not operate properly.

The PSEL.CLK and PSEL.DIN registers and their configurations are only used as long as the PDM module is enabled, and retained only as long as the device is in System ON mode. See POWER — Power supply on page 48 for more information about power modes. When the peripheral is disabled, the pins will behave as regular GPIOs, and use the configuration in their respective OUT bit field and PIN_CNF[n] register.

To ensure correct behaviour in the PDM module, the pins used by the PDM module must be configured in the GPIO peripheral as described in GPIO configuration before enabling peripheral on page 132 before enabling the PDM module. This is to ensure that the pins used by the PDM module are driven correctly if the PDM module itself is temporarily disabled or the device temporarily enters System OFF. This configuration must be retained in the GPIO for the selected I/Os as long as the PDM module is supposed to be connected to an external PDM circuit.

Only one peripheral can be assigned to drive a particular GPIO pin at a time. Failing to do so may result in unpredictable behaviour.

| PDM signal | PDM pin | Direction | Output value | Comment |
|---|---|---|---|---|
| CLK | As specified in PSEL.CLK | Output | 0 | |
| DIN | As specified in PSEL.DIN | Input | Not applicable | |

*Table 44: GPIO configuration before enabling peripheral*

## 6.10.7 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x4001D000 | PDM | PDM | Pulse-density modulation (digital microphone interface) | |

*Table 45: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_START | 0x000 | Starts continuous PDM transfer |
| TASKS_STOP | 0x004 | Stops PDM transfer |
| EVENTS_STARTED | 0x100 | PDM transfer has started |
| EVENTS_STOPPED | 0x104 | PDM transfer has finished |
| EVENTS_END | 0x108 | The PDM has written the last sample specified by SAMPLE.MAXCNT (or the last sample after a STOP task has been received) to Data RAM |
| INTEN | 0x300 | Enable or disable interrupt |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| ENABLE | 0x500 | PDM module enable register |
| PDMCLKCTRL | 0x504 | PDM clock generator control |
| MODE | 0x508 | Defines the routing of the connected PDM microphones' signals |
| GAINL | 0x518 | Left output gain adjustment |
| GAINR | 0x51C | Right output gain adjustment |
| PSEL.CLK | 0x540 | Pin number configuration for PDM CLK signal |
| PSEL.DIN | 0x544 | Pin number configuration for PDM DIN signal |
| SAMPLE.PTR | 0x560 | RAM address pointer to write samples to with EasyDMA |
| SAMPLE.MAXCNT | 0x564 | Number of samples to allocate memory for in EasyDMA mode |

*Table 46: Register overview*

## 6.10.7.1 TASKS_START

Address offset: 0x000

Starts continuous PDM transfer

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_START | | | Starts continuous PDM transfer |
| | | | Trigger | 1 | Trigger task |

## 6.10.7.2 TASKS_STOP

Address offset: 0x004

Stops PDM transfer

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STOP | | | Stops PDM transfer |
| | | | Trigger | 1 | Trigger task |

## 6.10.7.3 EVENTS_STARTED

Address offset: 0x100

PDM transfer has started

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_STARTED | | | PDM transfer has started |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.10.7.4 EVENTS_STOPPED

Address offset: 0x104

PDM transfer has finished

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_STOPPED | | | PDM transfer has finished |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.10.7.5 EVENTS_END

Address offset: 0x108

The PDM has written the last sample specified by SAMPLE.MAXCNT (or the last sample after a STOP task has been received) to Data RAM

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_END | | | The PDM has written the last sample specified by SAMPLE.MAXCNT (or the last sample after a STOP task has been received) to Data RAM |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.10.7.6 INTEN

Address offset: 0x300

Enable or disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | STARTED | | | Enable or disable interrupt for event STARTED |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| B | RW | STOPPED | | | Enable or disable interrupt for event STOPPED |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| C | RW | END | | | Enable or disable interrupt for event END |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |

## 6.10.7.7 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | STARTED | | | Write '1' to enable interrupt for event STARTED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | STOPPED | | | Write '1' to enable interrupt for event STOPPED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | END | | | Write '1' to enable interrupt for event END |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.10.7.8 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | STARTED | | | Write '1' to disable interrupt for event STARTED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | STOPPED | | | Write '1' to disable interrupt for event STOPPED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | END | | | Write '1' to disable interrupt for event END |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

### 6.10.7.9 ENABLE

Address offset: 0x500

PDM module enable register

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | ENABLE | | | Enable or disable PDM module |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |

### 6.10.7.10 PDMCLKCTRL

Address offset: 0x504

PDM clock generator control

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x08400000** | | | **0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | FREQ | | | PDM_CLK frequency |
| | | | 1000K | 0x08000000 | PDM_CLK = 32 MHz / 32 = 1.000 MHz |
| | | | Default | 0x08400000 | PDM_CLK = 32 MHz / 31 = 1.032 MHz |
| | | | 1067K | 0x08800000 | PDM_CLK = 32 MHz / 30 = 1.067 MHz |

### 6.10.7.11 MODE

Address offset: 0x508

Defines the routing of the connected PDM microphones' signals

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | B A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | OPERATION | | | Mono or stereo operation |
| | | | Stereo | 0 | Sample and store one pair (Left + Right) of 16bit samples per RAM word R=[31:16]; L=[15:0] |
| | | | Mono | 1 | Sample and store two successive Left samples (16 bit each) per RAM word L1=[31:16]; L0=[15:0] |
| B | RW | EDGE | | | Defines on which PDM_CLK edge Left (or mono) is sampled |
| | | | LeftFalling | 0 | Left (or mono) is sampled on falling edge of PDM_CLK |
| | | | LeftRising | 1 | Left (or mono) is sampled on rising edge of PDM_CLK |

### 6.10.7.12 GAINL

Address offset: 0x518

Left output gain adjustment

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A |
| **Reset 0x00000028** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | GAINL | | | Left output gain adjustment, in 0.5 dB steps, around the default module gain (see electrical parameters)<br>0x00 -20 dB gain adjust<br>0x01 -19.5 dB gain adjust<br>(...)<br>0x27 -0.5 dB gain adjust<br>0x28 0 dB gain adjust<br>0x29 +0.5 dB gain adjust<br>(...)<br>0x4F +19.5 dB gain adjust<br>0x50 +20 dB gain adjust |
| | | | MinGain | 0x00 | -20dB gain adjustment (minimum) |
| | | | DefaultGain | 0x28 | 0dB gain adjustment ('2500 RMS' requirement) |
| | | | MaxGain | 0x50 | +20dB gain adjustment (maximum) |

## 6.10.7.13 GAINR

Address offset: 0x51C

Right output gain adjustment

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A |
| **Reset 0x00000028** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | GAINR | | | Right output gain adjustment, in 0.5 dB steps, around the default module gain (see electrical parameters) |
| | | | MinGain | 0x00 | -20dB gain adjustment (minimum) |
| | | | DefaultGain | 0x28 | 0dB gain adjustment ('2500 RMS' requirement) |
| | | | MaxGain | 0x50 | +20dB gain adjustment (maximum) |

## 6.10.7.14 PSEL.CLK

Address offset: 0x540

Pin number configuration for PDM CLK signal

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | B A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| B | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

### 6.10.7.15 PSEL.DIN

Address offset: 0x544

Pin number configuration for PDM DIN signal

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | B A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| B | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

### 6.10.7.16 SAMPLE.PTR

Address offset: 0x560

RAM address pointer to write samples to with EasyDMA

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | SAMPLEPTR | | | Address to write PDM samples to over DMA |

### 6.10.7.17 SAMPLE.MAXCNT

Address offset: 0x564

Number of samples to allocate memory for in EasyDMA mode

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | BUFFSIZE | | [0..32767] | Length of DMA RAM allocation in number of samples |

## 6.10.8 Electrical specification

### 6.10.8.1 PDM Electrical Specification

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $f_{PDM,CLK}$ | PDM clock speed | | 1.032 | | MHz |
| $t_{PDM,JITTER}$ | Jitter in PDM clock output | | | 20 | ns |
| $T_{dPDM,CLK}$ | PDM clock duty cycle | 40 | 50 | 60 | % |
| $t_{PDM,DATA}$ | Decimation filter delay | | | 5 | ms |
| $t_{PDM,cv}$ | Allowed clock edge to data valid | | | 125 | ns |
| $t_{PDM,ci}$ | Allowed (other) clock edge to data invalid | 0 | | | ns |
| $t_{PDM,s}$ | Data setup time at $f_{PDM,CLK}$=1.024 MHz | 65 | | | ns |
| $t_{PDM,h}$ | Data hold time at $f_{PDM,CLK}$=1.024 MHz | 0 | | | ns |
| $G_{PDM,default}$ | Default (reset) absolute gain of the PDM module | | 3.2 | | dB |

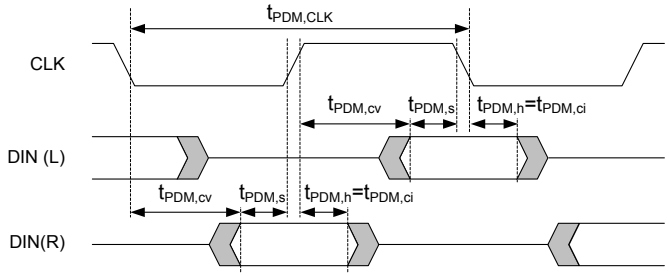



*Figure 45: PDM timing diagram*

## 6.11 PPI — Programmable peripheral interconnect

The programmable peripheral interconnect (PPI) enables peripherals to interact autonomously with each other using tasks and events independent of the CPU. The PPI allows precise synchronization between peripherals when real-time application constraints exist and eliminates the need for CPU activity to implement behavior which can be predefined using PPI.

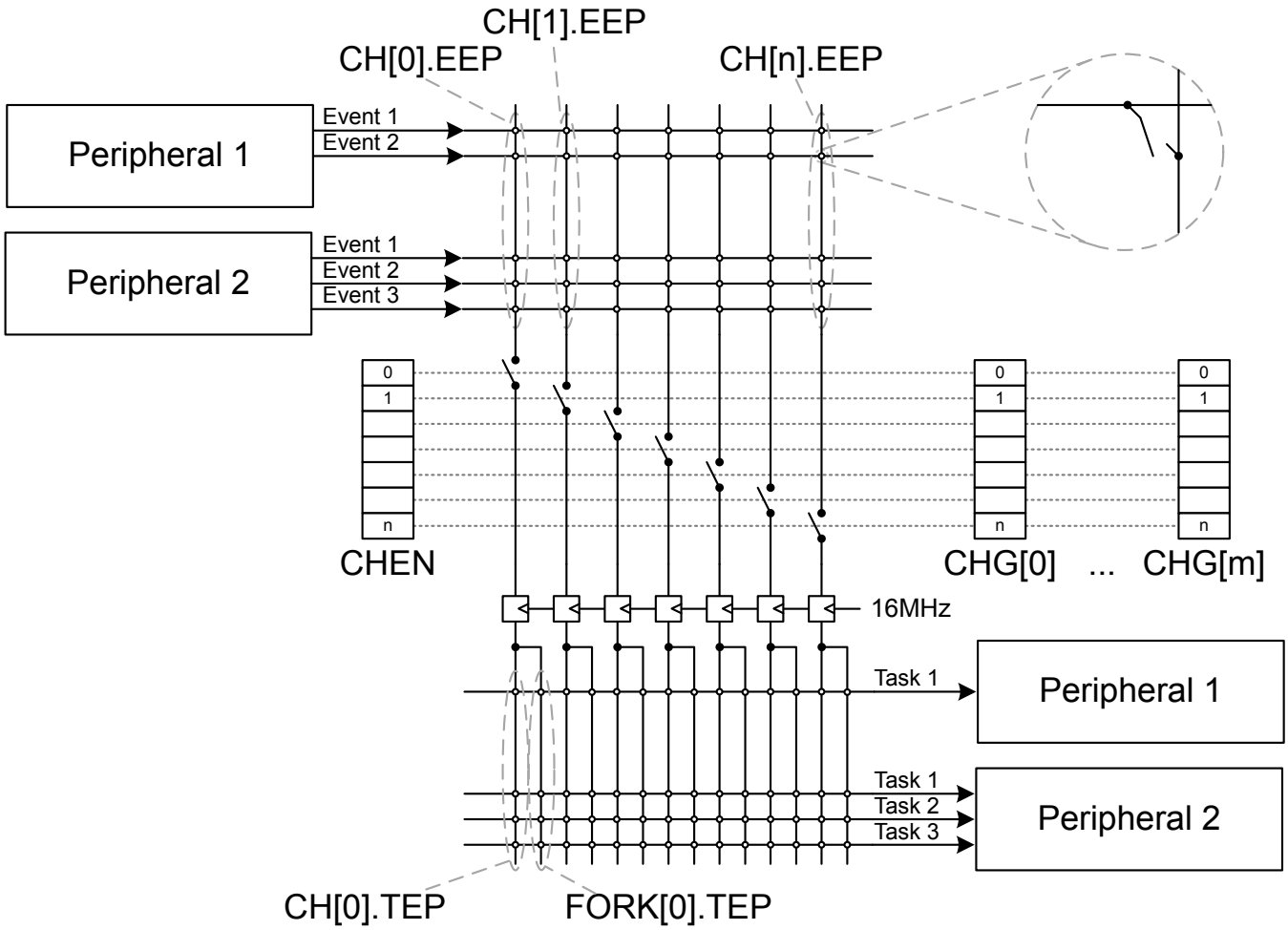



*Figure 46: PPI block diagram*

The PPI system has, in addition to the fully programmable peripheral interconnections, a set of channels where the event end point (EEP) and task end points (TEP) are fixed in hardware. These fixed channels can be individually enabled, disabled, or added to PPI channel groups (see CHG[n] registers), in the same way as ordinary PPI channels.

| Instance | Channel | Number of channels |
|---|---|---|
| PPI | 0-19 | 20 |
| PPI (fixed) | 20-31 | 12 |

*Table 47: Configurable and fixed PPI channels*

The PPI provides a mechanism to automatically trigger a task in one peripheral as a result of an event occurring in another peripheral. A task is connected to an event through a PPI channel. The PPI channel is composed of three end point registers, one EEP and two TEPs. A peripheral task is connected to a TEP using the address of the task register associated with the task. Similarly, a peripheral event is connected to an EEP using the address of the event register associated with the event.

On each PPI channel, the signals are synchronized to the 16 MHz clock, to avoid any internal violation of setup and hold timings. As a consequence, events that are synchronous to the 16 MHz clock will be delayed by one clock period, while other asynchronous events will be delayed by up to one 16 MHz clock period.

Note that shortcuts (as defined in the SHORTS register in each peripheral) are not affected by this 16 MHz synchronization, and are therefore not delayed.

Each TEP implements a fork mechanism that enables a second task to be triggered at the same time as the task specified in the TEP is triggered. This second task is configured in the task end point register in the FORK registers groups, e.g. FORK.TEP[0] is associated with PPI channel CH[0].

There are two ways of enabling and disabling PPI channels:

- Enable or disable PPI channels individually using the CHEN, CHENSET, and CHENCLR registers.
- Enable or disable PPI channels in PPI channel groups through the groups' ENABLE and DISABLE tasks. Prior to these tasks being triggered, the PPI channel group must be configured to define which PPI channels belong to which groups.

Note that when a channel belongs to two groups m and n, and the tasks CHG[m].EN and CHG[n].DIS occur simultaneously (m and n can be equal or different), the CHG[m].EN on that channel has priority.

PPI tasks (for example, CHG[0].EN) can be triggered through the PPI like any other task, which means they can be hooked to a PPI channel as a TEP. One event can trigger multiple tasks by using multiple channels and one task can be triggered by multiple events in the same way.

### 6.11.1 Pre-programmed channels

Some of the PPI channels are pre-programmed. These channels cannot be configured by the CPU, but can be added to groups and enabled and disabled like the general purpose PPI channels. The FORK TEP for these channels are still programmable and can be used by the application.

For a list of pre-programmed PPI channels, see the table below.

| Channel | EEP | TEP |
|---|---|---|
| 20 | TIMER0->EVENTS_COMPARE[0] | RADIO->TASKS_TXEN |
| 21 | TIMER0->EVENTS_COMPARE[0] | RADIO->TASKS_RXEN |
| 22 | TIMER0->EVENTS_COMPARE[1] | RADIO->TASKS_DISABLE |
| 23 | RADIO->EVENTS_BCMATCH | AAR->TASKS_START |
| 24 | RADIO->EVENTS_READY | CCM->TASKS_KSGEN |
| 25 | RADIO->EVENTS_ADDRESS | CCM->TASKS_CRYPT |
| 26 | RADIO->EVENTS_ADDRESS | TIMER0->TASKS_CAPTURE[1] |
| 27 | RADIO->EVENTS_END | TIMER0->TASKS_CAPTURE[2] |
| 28 | RTC0->EVENTS_COMPARE[0] | RADIO->TASKS_TXEN |
| 29 | RTC0->EVENTS_COMPARE[0] | RADIO->TASKS_RXEN |
| 30 | RTC0->EVENTS_COMPARE[0] | TIMER0->TASKS_CLEAR |
| 31 | RTC0->EVENTS_COMPARE[0] | TIMER0->TASKS_START |

*Table 48: Pre-programmed channels*

## 6.11.2 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x4001F000 | PPI | PPI | Programmable peripheral interconnect | |

*Table 49: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_CHG[0].EN | 0x000 | Enable channel group 0 |
| TASKS_CHG[0].DIS | 0x004 | Disable channel group 0 |
| TASKS_CHG[1].EN | 0x008 | Enable channel group 1 |
| TASKS_CHG[1].DIS | 0x00C | Disable channel group 1 |
| TASKS_CHG[2].EN | 0x010 | Enable channel group 2 |
| TASKS_CHG[2].DIS | 0x014 | Disable channel group 2 |
| TASKS_CHG[3].EN | 0x018 | Enable channel group 3 |
| TASKS_CHG[3].DIS | 0x01C | Disable channel group 3 |
| TASKS_CHG[4].EN | 0x020 | Enable channel group 4 |
| TASKS_CHG[4].DIS | 0x024 | Disable channel group 4 |
| TASKS_CHG[5].EN | 0x028 | Enable channel group 5 |
| TASKS_CHG[5].DIS | 0x02C | Disable channel group 5 |
| CHEN | 0x500 | Channel enable register |
| CHENSET | 0x504 | Channel enable set register |
| CHENCLR | 0x508 | Channel enable clear register |
| CH[0].EEP | 0x510 | Channel 0 event end-point |
| CH[0].TEP | 0x514 | Channel 0 task end-point |
| CH[1].EEP | 0x518 | Channel 1 event end-point |
| CH[1].TEP | 0x51C | Channel 1 task end-point |
| CH[2].EEP | 0x520 | Channel 2 event end-point |
| CH[2].TEP | 0x524 | Channel 2 task end-point |
| CH[3].EEP | 0x528 | Channel 3 event end-point |
| CH[3].TEP | 0x52C | Channel 3 task end-point |
| CH[4].EEP | 0x530 | Channel 4 event end-point |
| CH[4].TEP | 0x534 | Channel 4 task end-point |
| CH[5].EEP | 0x538 | Channel 5 event end-point |
| CH[5].TEP | 0x53C | Channel 5 task end-point |
| CH[6].EEP | 0x540 | Channel 6 event end-point |
| CH[6].TEP | 0x544 | Channel 6 task end-point |

| Register | Offset | Description |
|---|---|---|
| CH[7].EEP | 0x548 | Channel 7 event end-point |
| CH[7].TEP | 0x54C | Channel 7 task end-point |
| CH[8].EEP | 0x550 | Channel 8 event end-point |
| CH[8].TEP | 0x554 | Channel 8 task end-point |
| CH[9].EEP | 0x558 | Channel 9 event end-point |
| CH[9].TEP | 0x55C | Channel 9 task end-point |
| CH[10].EEP | 0x560 | Channel 10 event end-point |
| CH[10].TEP | 0x564 | Channel 10 task end-point |
| CH[11].EEP | 0x568 | Channel 11 event end-point |
| CH[11].TEP | 0x56C | Channel 11 task end-point |
| CH[12].EEP | 0x570 | Channel 12 event end-point |
| CH[12].TEP | 0x574 | Channel 12 task end-point |
| CH[13].EEP | 0x578 | Channel 13 event end-point |
| CH[13].TEP | 0x57C | Channel 13 task end-point |
| CH[14].EEP | 0x580 | Channel 14 event end-point |
| CH[14].TEP | 0x584 | Channel 14 task end-point |
| CH[15].EEP | 0x588 | Channel 15 event end-point |
| CH[15].TEP | 0x58C | Channel 15 task end-point |
| CH[16].EEP | 0x590 | Channel 16 event end-point |
| CH[16].TEP | 0x594 | Channel 16 task end-point |
| CH[17].EEP | 0x598 | Channel 17 event end-point |
| CH[17].TEP | 0x59C | Channel 17 task end-point |
| CH[18].EEP | 0x5A0 | Channel 18 event end-point |
| CH[18].TEP | 0x5A4 | Channel 18 task end-point |
| CH[19].EEP | 0x5A8 | Channel 19 event end-point |
| CH[19].TEP | 0x5AC | Channel 19 task end-point |
| CHG[0] | 0x800 | Channel group 0 |
| CHG[1] | 0x804 | Channel group 1 |
| CHG[2] | 0x808 | Channel group 2 |
| CHG[3] | 0x80C | Channel group 3 |
| CHG[4] | 0x810 | Channel group 4 |
| CHG[5] | 0x814 | Channel group 5 |
| FORK[0].TEP | 0x910 | Channel 0 task end-point |
| FORK[1].TEP | 0x914 | Channel 1 task end-point |
| FORK[2].TEP | 0x918 | Channel 2 task end-point |
| FORK[3].TEP | 0x91C | Channel 3 task end-point |
| FORK[4].TEP | 0x920 | Channel 4 task end-point |
| FORK[5].TEP | 0x924 | Channel 5 task end-point |
| FORK[6].TEP | 0x928 | Channel 6 task end-point |
| FORK[7].TEP | 0x92C | Channel 7 task end-point |
| FORK[8].TEP | 0x930 | Channel 8 task end-point |
| FORK[9].TEP | 0x934 | Channel 9 task end-point |
| FORK[10].TEP | 0x938 | Channel 10 task end-point |
| FORK[11].TEP | 0x93C | Channel 11 task end-point |
| FORK[12].TEP | 0x940 | Channel 12 task end-point |
| FORK[13].TEP | 0x944 | Channel 13 task end-point |
| FORK[14].TEP | 0x948 | Channel 14 task end-point |
| FORK[15].TEP | 0x94C | Channel 15 task end-point |
| FORK[16].TEP | 0x950 | Channel 16 task end-point |
| FORK[17].TEP | 0x954 | Channel 17 task end-point |
| FORK[18].TEP | 0x958 | Channel 18 task end-point |
| FORK[19].TEP | 0x95C | Channel 19 task end-point |
| FORK[20].TEP | 0x960 | Channel 20 task end-point |

| Register | Offset | Description |
|---|---|---|
| FORK[21].TEP | 0x964 | Channel 21 task end-point |
| FORK[22].TEP | 0x968 | Channel 22 task end-point |
| FORK[23].TEP | 0x96C | Channel 23 task end-point |
| FORK[24].TEP | 0x970 | Channel 24 task end-point |
| FORK[25].TEP | 0x974 | Channel 25 task end-point |
| FORK[26].TEP | 0x978 | Channel 26 task end-point |
| FORK[27].TEP | 0x97C | Channel 27 task end-point |
| FORK[28].TEP | 0x980 | Channel 28 task end-point |
| FORK[29].TEP | 0x984 | Channel 29 task end-point |
| FORK[30].TEP | 0x988 | Channel 30 task end-point |
| FORK[31].TEP | 0x98C | Channel 31 task end-point |

*Table 50: Register overview*

## 6.11.2.1 TASKS_CHG[n].EN (n=0..5)

Address offset: 0x000 + (n × 0x8)

Enable channel group n

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | EN | | | Enable channel group n |
| | | | Trigger | 1 | Trigger task |

## 6.11.2.2 TASKS_CHG[n].DIS (n=0..5)

Address offset: 0x004 + (n × 0x8)

Disable channel group n

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | DIS | | | Disable channel group n |
| | | | Trigger | 1 | Trigger task |

## 6.11.2.3 CHEN

Address offset: 0x500

Channel enable register

| Bit number | 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 |
|---|---|
| ID | f e d c b a Z Y X W V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A-f | RW | CH[i] (i=0..31) | | | Enable or disable channel i |
| | | | Disabled | 0 | Disable channel |
| | | | Enabled | 1 | Enable channel |

### 6.11.2.4 CHENSET

Address offset: 0x504

Channel enable set register

Read: reads value of CH{i} field in CHEN register.

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | f e d c b a Z Y X W V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A-f | RW | CH[i] (i=0..31) | | | Channel i enable set register. Writing '0' has no effect |
| | | | Disabled | 0 | Read: channel disabled |
| | | | Enabled | 1 | Read: channel enabled |
| | | | Set | 1 | Write: Enable channel |

### 6.11.2.5 CHENCLR

Address offset: 0x508

Channel enable clear register

Read: reads value of CH{i} field in CHEN register.

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | f e d c b a Z Y X W V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A-f | RW | CH[i] (i=0..31) | | | Channel i enable clear register. Writing '0' has no effect |
| | | | Disabled | 0 | Read: channel disabled |
| | | | Enabled | 1 | Read: channel enabled |
| | | | Clear | 1 | Write: disable channel |

### 6.11.2.6 CH[n].EEP (n=0..19)

Address offset: 0x510 + (n × 0x8)

Channel n event end-point

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | EEP | | | Pointer to event register. Accepts only addresses to registers from the Event group. |

### 6.11.2.7 CH[n].TEP (n=0..19)

Address offset: 0x514 + (n × 0x8)

Channel n task end-point

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | TEP | | | Pointer to task register. Accepts only addresses to registers from the Task group. |

### 6.11.2.8 CHG[n] (n=0..5)

Address offset: 0x800 + (n × 0x4)

Channel group n

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | f e d c b a Z Y X W V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-f | RW | CH[i] (i=0..31) | | | Include or exclude channel i |
| | | | Excluded | 0 | Exclude |
| | | | Included | 1 | Include |

### 6.11.2.9 FORK[n].TEP (n=0..31)

Address offset: 0x910 + (n × 0x4)

Channel n task end-point

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | TEP | | | Pointer to task register |

## 6.12 PWM — Pulse width modulation

The pulse with modulation (PWM) module enables the generation of pulse width modulated signals on GPIO. The module implements an up or up-and-down counter with four PWM channels that drive assigned GPIOs.

The following are the main features of a PWM module:

- Programmable PWM frequency
- Up to four PWM channels with individual polarity and duty cycle values
- Edge or center-aligned pulses across PWM channels
- Multiple duty cycle arrays (sequences) defined in RAM
- Autonomous and glitch-free update of duty cycle values directly from memory through EasyDMA (no CPU involvement)
- Change of polarity, duty cycle, and base frequency possibly on every PWM period
- RAM sequences can be repeated or connected into loops

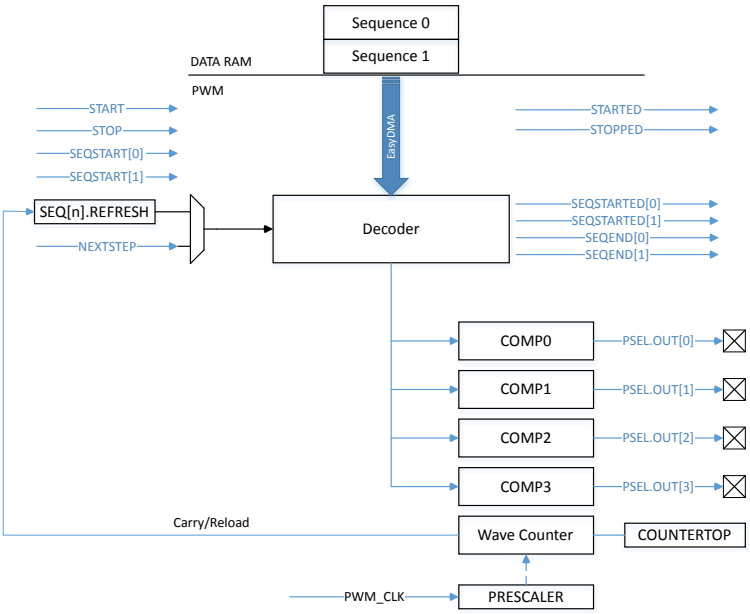



*Figure 47: PWM module*

## 6.12.1 Wave counter

The wave counter is responsible for generating the pulses at a duty cycle that depends on the compare values, and at a frequency that depends on COUNTERTOP.

There is one common 15-bit counter with four compare channels. Thus, all four channels will share the same period (PWM frequency), but can have individual duty cycle and polarity. The polarity is set by a value read from RAM (see figure Decoder memory access modes on page 149). Whether the counter counts up, or up and down, is controlled by the MODE register.

The timer top value is controlled by the COUNTERTOP register. This register value, in conjunction with the selected PRESCALER of the PWM_CLK, will result in a given PWM period. A COUNTERTOP value smaller than the compare setting will result in a state where no PWM edges are generated. OUT[n] is held high, given that the polarity is set to FallingEdge. All compare registers are internal and can only be configured through decoder presented later. COUNTERTOP can be safely written at any time.

Sampling follows the START task. If DECODER.LOAD=WaveForm, the register value is ignored and taken from RAM instead (see section Decoder with EasyDMA on page 149 for more details). If DECODER.LOAD is anything else than the WaveForm, it is sampled following a STARTSEQ[n] task and when loading a new value from RAM during a sequence playback.

The following figure shows the counter operating in up mode (MODE=PWM_MODE_Up), with three PWM channels with the same frequency but different duty cycle:

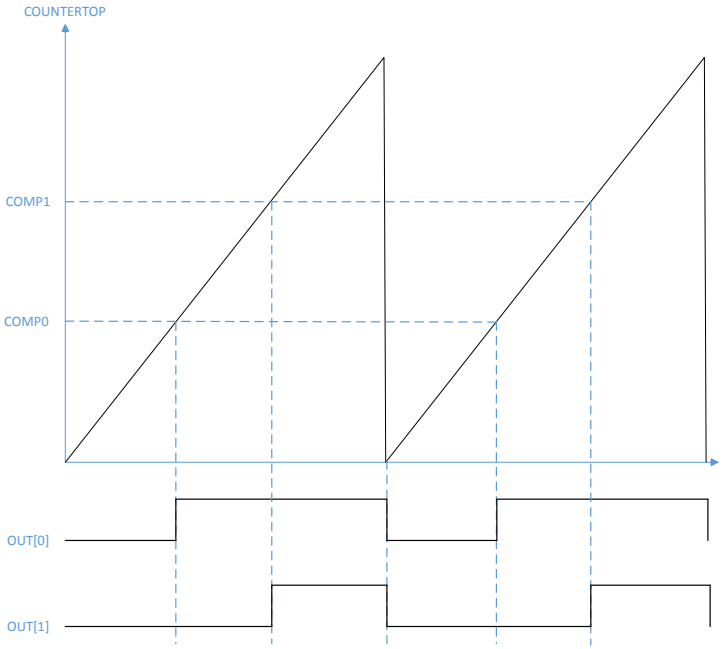



*Figure 48: PWM counter in up mode example - FallingEdge polarity*

The counter is automatically reset to zero when COUNTERTOP is reached and OUT[n] will invert. OUT[n] is held low if the compare value is 0 and held high if set to COUNTERTOP, given that the polarity is set to FallingEdge. Counter running in up mode results in pulse widths that are edge-aligned. The following is the code for the counter in up mode example:

```
uint16_t pwm_seq[4] = {PWM_CH0_DUTY, PWM_CH1_DUTY, PWM_CH2_DUTY, PWM_CH3_DUTY};
NRF_PWM0->PSEL.OUT[0] = (first_pin << PWM_PSEL_OUT_PIN_Pos) |
                        (PWM_PSEL_OUT_CONNECT_Connected <<
                                                PWM_PSEL_OUT_CONNECT_Pos);
NRF_PWM0->PSEL.OUT[1] = (second_pin << PWM_PSEL_OUT_PIN_Pos) |
                        (PWM_PSEL_OUT_CONNECT_Connected <<
                                                PWM_PSEL_OUT_CONNECT_Pos);
NRF_PWM0->ENABLE      = (PWM_ENABLE_ENABLE_Enabled << PWM_ENABLE_ENABLE_Pos);
NRF_PWM0->MODE        = (PWM_MODE_UPDOWN_Up << PWM_MODE_UPDOWN_Pos);
NRF_PWM0->PRESCALER   = (PWM_PRESCALER_PRESCALER_DIV_1 <<
                                                PWM_PRESCALER_PRESCALER_Pos);
NRF_PWM0->COUNTERTOP  = (16000 << PWM_COUNTERTOP_COUNTERTOP_Pos); //1 msec
NRF_PWM0->LOOP        = (PWM_LOOP_CNT_Disabled << PWM_LOOP_CNT_Pos);
NRF_PWM0->DECODER   = (PWM_DECODER_LOAD_Individual << PWM_DECODER_LOAD_Pos) |
                      (PWM_DECODER_MODE_RefreshCount << PWM_DECODER_MODE_Pos);
NRF_PWM0->SEQ[0].PTR  = ((uint32_t)(pwm_seq) << PWM_SEQ_PTR_PTR_Pos);
NRF_PWM0->SEQ[0].CNT  = ((sizeof(pwm_seq) / sizeof(uint16_t)) <<
                                                PWM_SEQ_CNT_CNT_Pos);
NRF_PWM0->SEQ[0].REFRESH  = 0;
NRF_PWM0->SEQ[0].ENDDELAY = 0;
NRF_PWM0->TASKS_SEQSTART[0] = 1;
```

When the counter is running in up mode, the following formula can be used to compute the PWM period and the step size:

PWM period: $T_{PWM(Up)} = T_{PWM\_CLK} * COUNTERTOP$

Step width/Resolution: $T_{steps} = T_{PWM\_CLK}$

The following figure shows the counter operating in up-and-down mode (MODE=PWM_MODE_UpAndDown), with two PWM channels with the same frequency but different duty cycle and output polarity:

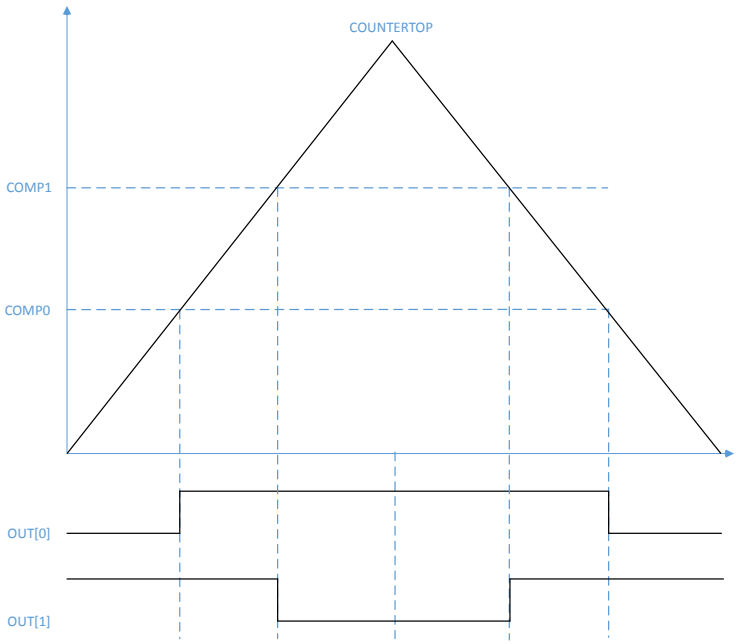



*Figure 49: PWM counter in up-and-down mode example*

The counter starts decrementing to zero when COUNTERTOP is reached and will invert the OUT[n] when compare value is hit for the second time. This results in a set of pulses that are center-aligned. The following is the code for the counter in up-and-down mode example:

```
uint16_t pwm_seq[4] = {PWM_CH0_DUTY, PWM_CH1_DUTY, PWM_CH2_DUTY, PWM_CH3_DUTY};
NRF_PWM0->PSEL.OUT[0] = (first_pin << PWM_PSEL_OUT_PIN_Pos) |
                        (PWM_PSEL_OUT_CONNECT_Connected <<
                                                PWM_PSEL_OUT_CONNECT_Pos);
NRF_PWM0->PSEL.OUT[1] = (second_pin << PWM_PSEL_OUT_PIN_Pos) |
                        (PWM_PSEL_OUT_CONNECT_Connected <<
                                                PWM_PSEL_OUT_CONNECT_Pos);
NRF_PWM0->ENABLE      = (PWM_ENABLE_ENABLE_Enabled << PWM_ENABLE_ENABLE_Pos);
NRF_PWM0->MODE        = (PWM_MODE_UPDOWN_UpAndDown << PWM_MODE_UPDOWN_Pos);
NRF_PWM0->PRESCALER   = (PWM_PRESCALER_PRESCALER_DIV_1 <<
                                                PWM_PRESCALER_PRESCALER_Pos);
NRF_PWM0->COUNTERTOP  = (16000 << PWM_COUNTERTOP_COUNTERTOP_Pos); //1 msec
NRF_PWM0->LOOP        = (PWM_LOOP_CNT_Disabled << PWM_LOOP_CNT_Pos);
NRF_PWM0->DECODER   = (PWM_DECODER_LOAD_Individual << PWM_DECODER_LOAD_Pos) |
                      (PWM_DECODER_MODE_RefreshCount << PWM_DECODER_MODE_Pos);
NRF_PWM0->SEQ[0].PTR  = ((uint32_t)(pwm_seq) << PWM_SEQ_PTR_PTR_Pos);
NRF_PWM0->SEQ[0].CNT  = ((sizeof(pwm_seq) / sizeof(uint16_t)) <<
                                                PWM_SEQ_CNT_CNT_Pos);
NRF_PWM0->SEQ[0].REFRESH  = 0;
NRF_PWM0->SEQ[0].ENDDELAY = 0;
NRF_PWM0->TASKS_SEQSTART[0] = 1;
```

When the counter is running in up-and-down mode, the following formula can be used to compute the PWM period and the step size:

$T_{PWM(Up\ And\ Down)} = T_{PWM\_CLK} * 2 * COUNTERTOP$

Step width/Resolution: $T_{steps} = T_{PWM\_CLK} * 2$

## 6.12.2 Decoder with EasyDMA

The decoder uses EasyDMA to take PWM parameters stored in RAM and update the internal compare registers of the wave counter, based on the mode of operation.

PWM parameters are organized into a sequence containing at least one half word (16 bit). Its most significant bit[15] denotes the polarity of the OUT[n] while bit[14:0] is the 15-bit compare value.

| Bit number | | | | 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 |
|---|---|---|---|---|
| Id | | | | B A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| Id | RW | Field | Value Id | Value | Description |
| A | RW | COMPARE | | | Duty cycle setting - value loaded to internal compare register |
| B | RW | POLARITY | | | Edge polarity of GPIO. |
| | | | RisingEdge | 0 | First edge within the PWM period is rising |
| | | | FallingEdge | 1 | First edge within the PWM period is falling |

The DECODER register controls how the RAM content is interpreted and loaded into the internal compare registers. The LOAD field controls if the RAM values are loaded to all compare channels, or to update a group or all channels with individual values. The following figure illustrates how parameters stored in RAM are organized and routed to various compare channels in different modes:

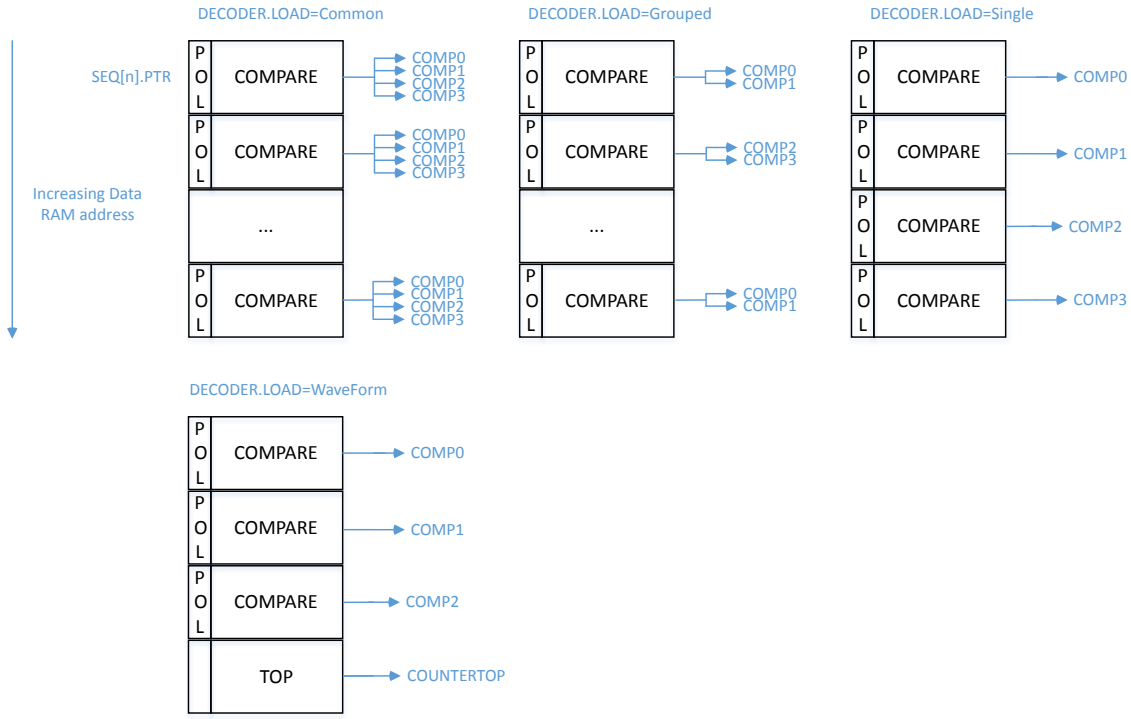



*Figure 50: Decoder memory access modes*

A special mode of operation is available when DECODER.LOAD is set to WaveForm. In this mode, up to three PWM channels can be enabled - OUT[0] to OUT[2]. In RAM, four values are loaded at a time: the first, second and third location are used to load the values, and the fourth RAM location is used to load

the COUNTERTOP register. This way one can have up to three PWM channels with a frequency base that changes on a per PWM period basis. This mode of operation is useful for arbitrary wave form generation in applications, such as LED lighting.

The register SEQ[n].REFRESH=N (one per sequence n=0 or 1) will instruct a new RAM stored pulse width value on every (N+1)$^{th}$ PWM period. Setting the register to zero will result in a new duty cycle update every PWM period, as long as the minimum PWM period is observed.

Note that registers SEQ[n].REFRESH and SEQ[n].ENDDELAY are ignored when DECODER.MODE=NextStep. The next value is loaded upon every received NEXTSTEP task.

SEQ[n].PTR is the pointer used to fetch COMPARE values from RAM. If the SEQ[n].PTR is not pointing to a RAM region, an EasyDMA transfer may result in a HardFault or RAM corruption. See Memory on page 16 for more information about the different memory regions. After the SEQ[n].PTR is set to the desired RAM location, the SEQ[n].CNT register must be set to number of 16-bit half words in the sequence. It is important to observe that the Grouped mode requires one half word per group, while the Single mode requires one half word per channel, thus increasing the RAM size occupation. If PWM generation is not running when the SEQSTART[n] task is triggered, the task will load the first value from RAM and then start the PWM generation. A SEQSTARTED[n] event is generated as soon as the EasyDMA has read the first PWM parameter from RAM and the wave counter has started executing it. When LOOP.CNT=0, sequence n=0 or 1 is played back once. After the last value in the sequence has been loaded and started executing, a SEQEND[n] event is generated. The PWM generation will then continue with the last loaded value. The following figure illustrates an example of such simple playback:

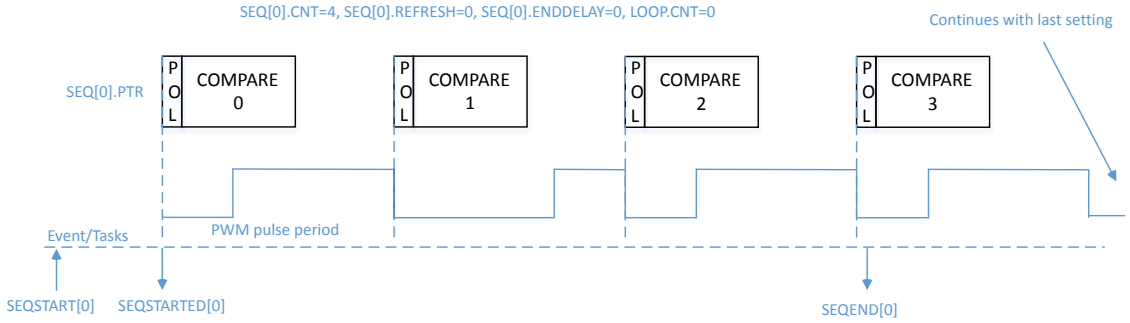



*Figure 51: Simple sequence example*

Figure depicts the source code used for configuration and timing details in a sequence where only sequence 0 is used and only run once with a new PWM duty cycle for each period.

```
NRF_PWM0->PSEL.OUT[0] = (first_pin << PWM_PSEL_OUT_PIN_Pos) |
                        (PWM_PSEL_OUT_CONNECT_Connected <<
                                                PWM_PSEL_OUT_CONNECT_Pos);
NRF_PWM0->ENABLE      = (PWM_ENABLE_ENABLE_Enabled << PWM_ENABLE_ENABLE_Pos);
NRF_PWM0->MODE        = (PWM_MODE_UPDOWN_Up << PWM_MODE_UPDOWN_Pos);
NRF_PWM0->PRESCALER   = (PWM_PRESCALER_PRESCALER_DIV_1 <<
                                                PWM_PRESCALER_PRESCALER_Pos);
NRF_PWM0->COUNTERTOP  = (16000 << PWM_COUNTERTOP_COUNTERTOP_Pos); //1 msec
NRF_PWM0->LOOP        = (PWM_LOOP_CNT_Disabled << PWM_LOOP_CNT_Pos);
NRF_PWM0->DECODER   = (PWM_DECODER_LOAD_Common << PWM_DECODER_LOAD_Pos) |
                      (PWM_DECODER_MODE_RefreshCount << PWM_DECODER_MODE_Pos);
NRF_PWM0->SEQ[0].PTR  = ((uint32_t)(seq0_ram) << PWM_SEQ_PTR_PTR_Pos);
NRF_PWM0->SEQ[0].CNT  = ((sizeof(seq0_ram) / sizeof(uint16_t)) <<
                                                PWM_SEQ_CNT_CNT_Pos);
NRF_PWM0->SEQ[0].REFRESH  = 0;
NRF_PWM0->SEQ[0].ENDDELAY = 0;
NRF_PWM0->TASKS_SEQSTART[0] = 1;
```

To completely stop the PWM generation and force the associated pins to a defined state, a STOP task can be triggered at any time. A STOPPED event is generated when the PWM generation has stopped at the end of currently running PWM period, and the pins go into their idle state as defined in GPIO OUT register. PWM generation can then only be restarted through a SEQSTART[n] task. SEQSTART[n] will resume PWM generation after having loaded the first value from the RAM buffer defined in the SEQ[n].PTR register.

The table below indicates when specific registers get sampled by the hardware. Care should be taken when updating these registers to avoid that values are applied earlier than expected.

| Register | Taken into account by hardware | Recommended (safe) update |
|---|---|---|
| SEQ[n].PTR | When sending the SEQSTART[n] task | After having received the SEQSTARTED[n] event |
| SEQ[n].CNT | When sending the SEQSTART[n] task | After having received the SEQSTARTED[n] event |
| SEQ[0].ENDDELAY | When sending the SEQSTART[0] task | Before starting sequence [0] through a SEQSTART[0] task |
| | Every time a new value from sequence [0] has been loaded from RAM and gets applied to the Wave Counter (indicated by the PWMPERIODEND event) | When no more value from sequence [0] gets loaded from RAM (indicated by the SEQEND[0] event) |
| | | At any time during sequence [1] (which starts when the SEQSTARTED[1] event is generated) |
| SEQ[1].ENDDELAY | When sending the SEQSTART[1] task | Before starting sequence [1] through a SEQSTART[1] task |
| | Every time a new value from sequence [1] has been loaded from RAM and gets applied to the Wave Counter (indicated by the PWMPERIODEND event) | When no more value from sequence [1] gets loaded from RAM (indicated by the SEQEND[1] event) |
| | | At any time during sequence [0] (which starts when the SEQSTARTED[0] event is generated) |
| SEQ[0].REFRESH | When sending the SEQSTART[0] task | Before starting sequence [0] through a SEQSTART[0] task |
| | Every time a new value from sequence [0] has been loaded from RAM and gets applied to the Wave Counter (indicated by the PWMPERIODEND event) | At any time during sequence [1] (which starts when the SEQSTARTED[1] event is generated) |
| SEQ[1].REFRESH | When sending the SEQSTART[1] task | Before starting sequence [1] through a SEQSTART[1] task |
| | Every time a new value from sequence [1] has been loaded from RAM and gets applied to the Wave Counter (indicated by the PWMPERIODEND event) | At any time during sequence [0] (which starts when the SEQSTARTED[0] event is generated) |
| COUNTERTOP | In DECODER.LOAD=WaveForm: this register is ignored. | Before starting PWM generation through a SEQSTART[n] task |
| | In all other LOAD modes: at the end of current PWM period (indicated by the PWMPERIODEND event) | After a STOP task has been triggered, and the STOPPED event has been received. |
| MODE | Immediately | Before starting PWM generation through a SEQSTART[n] task |
| | | After a STOP task has been triggered, and the STOPPED event has been received. |
| DECODER | Immediately | Before starting PWM generation through a SEQSTART[n] task |
| | | After a STOP task has been triggered, and the STOPPED event has been received. |
| PRESCALER | Immediately | Before starting PWM generation through a SEQSTART[n] task |
| | | After a STOP task has been triggered, and the STOPPED event has been received. |
| LOOP | Immediately | Before starting PWM generation through a SEQSTART[n] task |
| | | After a STOP task has been triggered, and the STOPPED event has been received. |
| PSEL.OUT[n] | Immediately | Before enabling the PWM instance through the ENABLE register |

*Table 51: When to safely update PWM registers*

**Note:** SEQ[n].REFRESH and SEQ[n].ENDDELAY are ignored at the end of a complex sequence, indicated by a LOOPSDONE event. The reason for this is that the last value loaded from RAM is maintained until further action from software (restarting a new sequence, or stopping PWM generation).

A more complex example, where LOOP.CNT>0, is shown in the following figure:

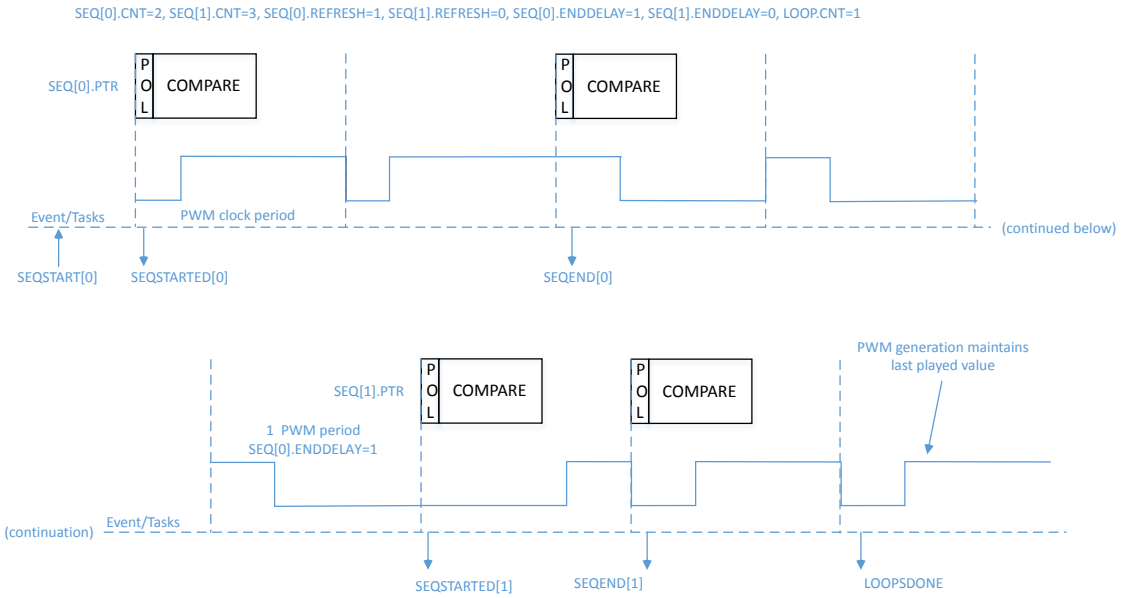



*Figure 52: Example using two sequences*

In this case, an automated playback takes place, consisting of SEQ[0], delay 0, SEQ[1], delay 1, then again SEQ[0], etc. The user can choose to start a complex playback with SEQ[0] or SEQ[1] through sending the SEQSTART[0] or SEQSTART[1] task. The complex playback always ends with delay 1.

The two sequences 0 and 1 are defined by the addresses of value tables in RAM (pointed to by SEQ[n].PTR) and the buffer size (SEQ[n].CNT). The rate at which a new value is loaded is defined individually for each sequence by SEQ[n].REFRESH. The chaining of sequence 1 following the sequence 0 is implicit, the LOOP.CNT register allows the chaining of sequence 1 to sequence 0 for a determined number of times. In other words, it allows to repeat a complex sequence a number of times in a fully automated way.

In the following code example, sequence 0 is defined with SEQ[0].REFRESH set to 1, meaning that a new PWM duty cycle is pushed every second PWM period. This complex sequence is started with the SEQSTART[0] task, so SEQ[0] is played first. Since SEQ[0].ENDDELAY=1 there will be one PWM period delay between last period on sequence 0 and the first period on sequence 1. Since SEQ[1].ENDDELAY=0 there is no delay 1, so SEQ[0] would be started immediately after the end of SEQ[1]. However, as LOOP.CNT is

1, the playback stops after having played SEQ[1] only once, and both SEQEND[1] and LOOPSDONE are generated (their order is not guaranteed in this case).

```
NRF_PWM0->PSEL.OUT[0] = (first_pin << PWM_PSEL_OUT_PIN_Pos) |
                        (PWM_PSEL_OUT_CONNECT_Connected <<
                                                  PWM_PSEL_OUT_CONNECT_Pos);
NRF_PWM0->ENABLE      = (PWM_ENABLE_ENABLE_Enabled << PWM_ENABLE_ENABLE_Pos);
NRF_PWM0->MODE        = (PWM_MODE_UPDOWN_Up << PWM_MODE_UPDOWN_Pos);
NRF_PWM0->PRESCALER   = (PWM_PRESCALER_PRESCALER_DIV_1 <<
                                                  PWM_PRESCALER_PRESCALER_Pos);
NRF_PWM0->COUNTERTOP  = (16000 << PWM_COUNTERTOP_COUNTERTOP_Pos); //1 msec
NRF_PWM0->LOOP        = (1 << PWM_LOOP_CNT_Pos);
NRF_PWM0->DECODER   = (PWM_DECODER_LOAD_Common << PWM_DECODER_LOAD_Pos) |
                      (PWM_DECODER_MODE_RefreshCount << PWM_DECODER_MODE_Pos);
NRF_PWM0->SEQ[0].PTR  = ((uint32_t)(seq0_ram) << PWM_SEQ_PTR_PTR_Pos);
NRF_PWM0->SEQ[0].CNT  = ((sizeof(seq0_ram) / sizeof(uint16_t)) <<
                                                  PWM_SEQ_CNT_CNT_Pos);
NRF_PWM0->SEQ[0].REFRESH  = 1;
NRF_PWM0->SEQ[0].ENDDELAY = 1;
NRF_PWM0->SEQ[1].PTR  = ((uint32_t)(seq1_ram) << PWM_SEQ_PTR_PTR_Pos);
NRF_PWM0->SEQ[1].CNT  = ((sizeof(seq1_ram) / sizeof(uint16_t)) <<
                                                  PWM_SEQ_CNT_CNT_Pos);
NRF_PWM0->SEQ[1].REFRESH  = 0;
NRF_PWM0->SEQ[1].ENDDELAY = 0;
NRF_PWM0->TASKS_SEQSTART[0] = 1;
```

The decoder can also be configured to asynchronously load new PWM duty cycle. If the DECODER.MODE register is set to NextStep, then the NEXTSTEP task will cause an update of internal compare registers on the next PWM period.

The following figures provide an overview of each part of an arbitrary sequence, in various modes (LOOP.CNT=0 and LOOP.CNT>0). In particular, the following are represented:

- Initial and final duty cycle on the PWM output(s)
- Chaining of SEQ[0] and SEQ[1] if LOOP.CNT>0
- Influence of registers on the sequence
- Events generated during a sequence
- DMA activity (loading of next value and applying it to the output(s))

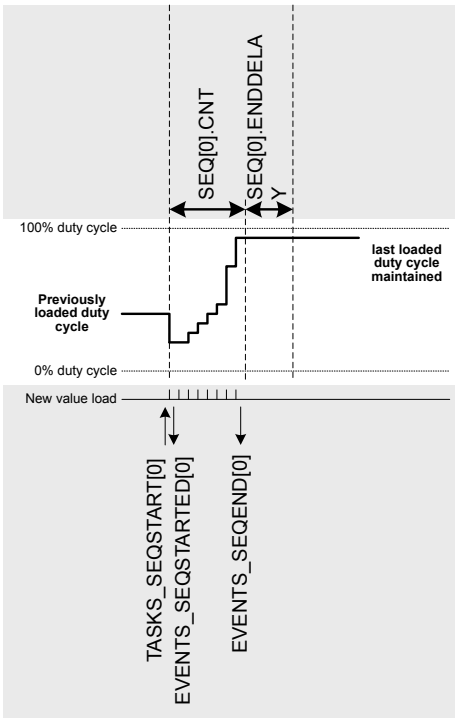



*Figure 53: Single shot (LOOP.CNT=0)*

**Note:** The single-shot example also applies to SEQ[1]. Only SEQ[0] is represented for simplicity.

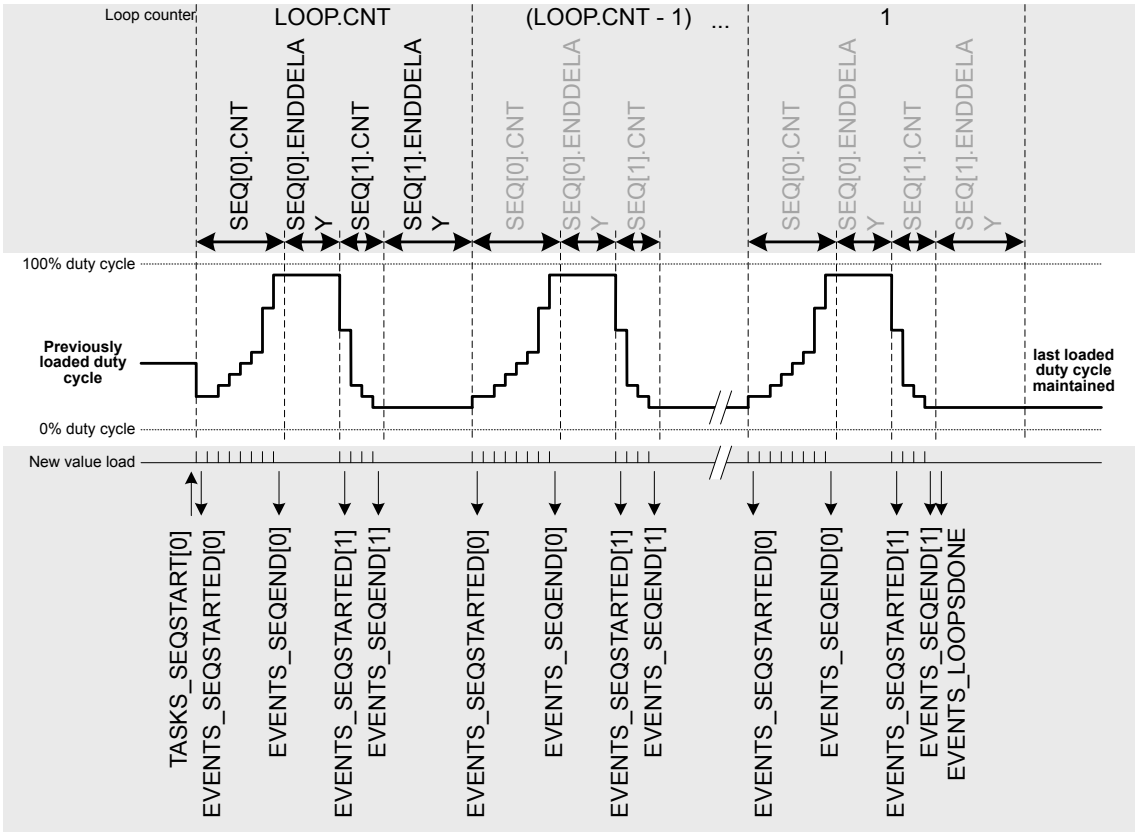



*Figure 54: Complex sequence (LOOP.CNT>0) starting with SEQ[0]*

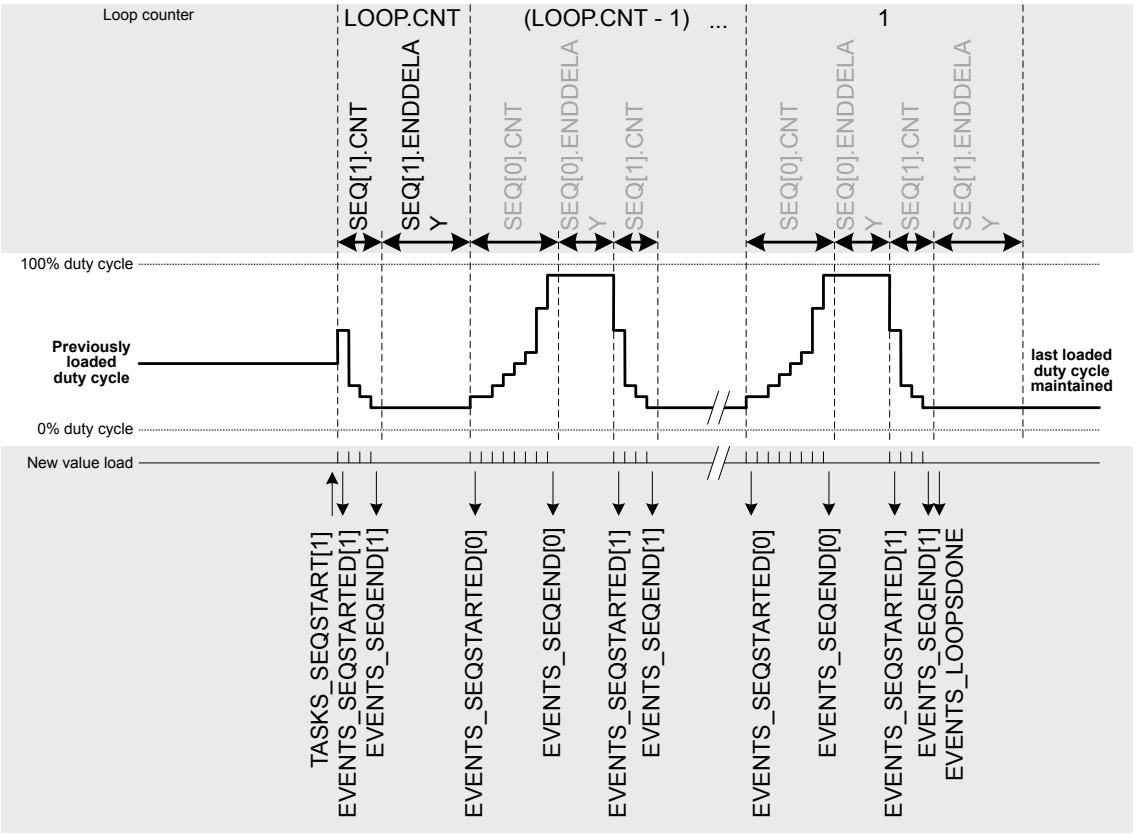



*Figure 55: Complex sequence (LOOP.CNT>0) starting with SEQ[1]*

**Note:** If a sequence is in use in a simple or complex sequence, it must have a length of SEQ[n].CNT > 0.

## 6.12.3 Limitations

Previous compare value is repeated if the PWM period is shorter than the time it takes for the EasyDMA to retrieve from RAM and update the internal compare registers. This is to ensure a glitch-free operation even for very short PWM periods.

## 6.12.4 Pin configuration

The OUT[n] (n=0..3) signals associated with each PWM channel are mapped to physical pins according to the configuration of PSEL.OUT[n] registers. If PSEL.OUT[n].CONNECT is set to Disconnected, the associated PWM module signal will not be connected to any physical pins.

The PSEL.OUT[n] registers and their configurations are used as long as the PWM module is enabled and the PWM generation active (wave counter started). They are retained only as long as the device is in System ON mode (see section POWER for more information about power modes).

To ensure correct behavior in the PWM module, the pins that are used must be configured in the GPIO peripheral in the following way before the PWM module is enabled:

| PWM signal | PWM pin | Direction | Output value | Comment |
|---|---|---|---|---|
| OUT[n] | As specified in PSEL.OUT[n] (n=0..3) | Output | 0 | Idle state defined in GPIO OUT register |

*Table 52: Recommended GPIO configuration before starting PWM generation*

The idle state of a pin is defined by the OUT register in the GPIO module, to ensure that the pins used by the PWM module are driven correctly. If PWM generation is stopped by triggering a STOP task, the PWM module itself is temporarily disabled or the device temporarily enters System OFF. This configuration must be retained in the GPIO for the selected pins (I/Os) for as long as the PWM module is supposed to be connected to an external PWM circuit.

Only one peripheral can be assigned to drive a particular GPIO pin at a time. Failing to do so may result in unpredictable behavior.

## 6.12.5 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x4001C000 | PWM | PWM0 | Pulse-width modulation unit 0 | |

*Table 53: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_STOP | 0x004 | Stops PWM pulse generation on all channels at the end of current PWM period, and stops sequence playback |
| TASKS_SEQSTART[0] | 0x008 | Loads the first PWM value on all enabled channels from sequence 0, and starts playing that sequence at the rate defined in SEQ[0]REFRESH and/or DECODER.MODE. Causes PWM generation to start if not running. |
| TASKS_SEQSTART[1] | 0x00C | Loads the first PWM value on all enabled channels from sequence 1, and starts playing that sequence at the rate defined in SEQ[1]REFRESH and/or DECODER.MODE. Causes PWM generation to start if not running. |
| TASKS_NEXTSTEP | 0x010 | Steps by one value in the current sequence on all enabled channels if DECODER.MODE=NextStep. Does not cause PWM generation to start if not running. |
| EVENTS_STOPPED | 0x104 | Response to STOP task, emitted when PWM pulses are no longer generated |
| EVENTS_SEQSTARTED[0] | 0x108 | First PWM period started on sequence 0 |
| EVENTS_SEQSTARTED[1] | 0x10C | First PWM period started on sequence 1 |
| EVENTS_SEQEND[0] | 0x110 | Emitted at end of every sequence 0, when last value from RAM has been applied to wave counter |
| EVENTS_SEQEND[1] | 0x114 | Emitted at end of every sequence 1, when last value from RAM has been applied to wave counter |
| EVENTS_PWMPERIODEND | 0x118 | Emitted at the end of each PWM period |
| EVENTS_LOOPSDONE | 0x11C | Concatenated sequences have been played the amount of times defined in LOOP.CNT |
| SHORTS | 0x200 | Shortcuts between local events and tasks |
| INTEN | 0x300 | Enable or disable interrupt |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| ENABLE | 0x500 | PWM module enable register |
| MODE | 0x504 | Selects operating mode of the wave counter |
| COUNTERTOP | 0x508 | Value up to which the pulse generator counter counts |
| PRESCALER | 0x50C | Configuration for PWM_CLK |
| DECODER | 0x510 | Configuration of the decoder |
| LOOP | 0x514 | Number of playbacks of a loop |
| SEQ[0].PTR | 0x520 | Beginning address in RAM of this sequence |
| SEQ[0].CNT | 0x524 | Number of values (duty cycles) in this sequence |
| SEQ[0].REFRESH | 0x528 | Number of additional PWM periods between samples loaded into compare register |
| SEQ[0].ENDDELAY | 0x52C | Time added after the sequence |
| SEQ[1].PTR | 0x540 | Beginning address in RAM of this sequence |
| SEQ[1].CNT | 0x544 | Number of values (duty cycles) in this sequence |
| SEQ[1].REFRESH | 0x548 | Number of additional PWM periods between samples loaded into compare register |
| SEQ[1].ENDDELAY | 0x54C | Time added after the sequence |

| Register | Offset | Description |
|---|---|---|
| PSEL.OUT[0] | 0x560 | Output pin select for PWM channel 0 |
| PSEL.OUT[1] | 0x564 | Output pin select for PWM channel 1 |
| PSEL.OUT[2] | 0x568 | Output pin select for PWM channel 2 |
| PSEL.OUT[3] | 0x56C | Output pin select for PWM channel 3 |

*Table 54: Register overview*

### 6.12.5.1 TASKS_STOP

Address offset: 0x004

Stops PWM pulse generation on all channels at the end of current PWM period, and stops sequence playback

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STOP | | | Stops PWM pulse generation on all channels at the end of current PWM period, and stops sequence playback |
| | | | Trigger | 1 | Trigger task |

### 6.12.5.2 TASKS_SEQSTART[n] (n=0..1)

Address offset: 0x008 + (n × 0x4)

Loads the first PWM value on all enabled channels from sequence n, and starts playing that sequence at the rate defined in SEQ[n]REFRESH and/or DECODER.MODE. Causes PWM generation to start if not running.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_SEQSTART | | | Loads the first PWM value on all enabled channels from sequence n, and starts playing that sequence at the rate defined in SEQ[n]REFRESH and/or DECODER.MODE. Causes PWM generation to start if not running. |
| | | | Trigger | 1 | Trigger task |

### 6.12.5.3 TASKS_NEXTSTEP

Address offset: 0x010

Steps by one value in the current sequence on all enabled channels if DECODER.MODE=NextStep. Does not cause PWM generation to start if not running.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_NEXTSTEP | | | Steps by one value in the current sequence on all enabled channels if DECODER.MODE=NextStep. Does not cause PWM generation to start if not running. |
| | | | Trigger | 1 | Trigger task |

## 6.12.5.4 EVENTS_STOPPED

Address offset: 0x104

Response to STOP task, emitted when PWM pulses are no longer generated

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_STOPPED | | | Response to STOP task, emitted when PWM pulses are no longer generated |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.12.5.5 EVENTS_SEQSTARTED[n] (n=0..1)

Address offset: 0x108 + (n × 0x4)

First PWM period started on sequence n

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_SEQSTARTED | | | First PWM period started on sequence n |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.12.5.6 EVENTS_SEQEND[n] (n=0..1)

Address offset: 0x110 + (n × 0x4)

Emitted at end of every sequence n, when last value from RAM has been applied to wave counter

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_SEQEND | | | Emitted at end of every sequence n, when last value from RAM has been applied to wave counter |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.12.5.7 EVENTS_PWMPERIODEND

Address offset: 0x118

Emitted at the end of each PWM period

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_PWMPERIODEND | | | Emitted at the end of each PWM period |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.12.5.8 EVENTS_LOOPSDONE

Address offset: 0x11C

Concatenated sequences have been played the amount of times defined in LOOP.CNT

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_LOOPSDONE | | | Concatenated sequences have been played the amount of times defined in LOOP.CNT |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.12.5.9 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | E D C B A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | SEQEND0_STOP | | | Shortcut between event SEQEND[0] and task STOP |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| B | RW | SEQEND1_STOP | | | Shortcut between event SEQEND[1] and task STOP |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| C | RW | LOOPSDONE_SEQSTART0 | | | Shortcut between event LOOPSDONE and task SEQSTART[0] |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| D | RW | LOOPSDONE_SEQSTART1 | | | Shortcut between event LOOPSDONE and task SEQSTART[1] |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| E | RW | LOOPSDONE_STOP | | | Shortcut between event LOOPSDONE and task STOP |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |

## 6.12.5.10 INTEN

Address offset: 0x300

Enable or disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | H G F E D C B |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| B | RW | STOPPED | | | Enable or disable interrupt for event STOPPED |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| C-D | RW | SEQSTARTED[i] (i=0..1) | | | Enable or disable interrupt for event SEQSTARTED[i] |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| E-F | RW | SEQEND[i] (i=0..1) | | | Enable or disable interrupt for event SEQEND[i] |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| G | RW | PWMPERIODEND | | | Enable or disable interrupt for event PWMPERIODEND |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| H | RW | LOOPSDONE | | | Enable or disable interrupt for event LOOPSDONE |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |

## 6.12.5.11 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | H G F E D C B |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| B | RW | STOPPED | | | Write '1' to enable interrupt for event STOPPED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C-D | RW | SEQSTARTED[i] (i=0..1) | | | Write '1' to enable interrupt for event SEQSTARTED[i] |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E-F | RW | SEQEND[i] (i=0..1) | | | Write '1' to enable interrupt for event SEQEND[i] |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| G | RW | PWMPERIODEND | | | Write '1' to enable interrupt for event PWMPERIODEND |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| H | RW | LOOPSDONE | | | Write '1' to enable interrupt for event LOOPSDONE |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |

| Bit number | 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 |
|---|---|
| ID | H G F E D C B |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| | | | Enabled | 1 | Read: Enabled |

## 6.12.5.12 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | 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 |
|---|---|
| ID | H G F E D C B |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| B | RW | STOPPED | | | Write '1' to disable interrupt for event STOPPED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C-D | RW | SEQSTARTED[i] (i=0..1) | | | Write '1' to disable interrupt for event SEQSTARTED[i] |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E-F | RW | SEQEND[i] (i=0..1) | | | Write '1' to disable interrupt for event SEQEND[i] |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| G | RW | PWMPERIODEND | | | Write '1' to disable interrupt for event PWMPERIODEND |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| H | RW | LOOPSDONE | | | Write '1' to disable interrupt for event LOOPSDONE |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.12.5.13 ENABLE

Address offset: 0x500

PWM module enable register

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | ENABLE | | | Enable or disable PWM module |
| | | | Disabled | 0 | Disabled |
| | | | Enabled | 1 | Enable |

## 6.12.5.14 MODE

Address offset: 0x504

Selects operating mode of the wave counter

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | UPDOWN | | | Selects up mode or up-and-down mode for the counter |
| | | | Up | 0 | Up counter, edge-aligned PWM duty cycle |
| | | | UpAndDown | 1 | Up and down counter, center-aligned PWM duty cycle |

## 6.12.5.15 COUNTERTOP

Address offset: 0x508

Value up to which the pulse generator counter counts

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A |
| **Reset 0x000003FF** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | COUNTERTOP | | [3..32767] | Value up to which the pulse generator counter counts. This register is ignored when DECODER.MODE=WaveForm and only values from RAM are used. |

## 6.12.5.16 PRESCALER

Address offset: 0x50C

Configuration for PWM_CLK

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PRESCALER | | | Prescaler of PWM_CLK |
| | | | DIV_1 | 0 | Divide by 1 (16 MHz) |
| | | | DIV_2 | 1 | Divide by 2 (8 MHz) |
| | | | DIV_4 | 2 | Divide by 4 (4 MHz) |
| | | | DIV_8 | 3 | Divide by 8 (2 MHz) |
| | | | DIV_16 | 4 | Divide by 16 (1 MHz) |
| | | | DIV_32 | 5 | Divide by 32 (500 kHz) |
| | | | DIV_64 | 6 | Divide by 64 (250 kHz) |
| | | | DIV_128 | 7 | Divide by 128 (125 kHz) |

## 6.12.5.17 DECODER

Address offset: 0x510

Configuration of the decoder

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | B A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | LOAD | | | How a sequence is read from RAM and spread to the compare register |
| | | | Common | 0 | 1st half word (16-bit) used in all PWM channels 0..3 |
| | | | Grouped | 1 | 1st half word (16-bit) used in channel 0..1; 2nd word in channel 2..3 |
| | | | Individual | 2 | 1st half word (16-bit) in ch.0; 2nd in ch.1; ...; 4th in ch.3 |
| | | | WaveForm | 3 | 1st half word (16-bit) in ch.0; 2nd in ch.1; ...; 4th in COUNTERTOP |
| B | RW | MODE | | | Selects source for advancing the active sequence |
| | | | RefreshCount | 0 | SEQ[n].REFRESH is used to determine loading internal compare registers |
| | | | NextStep | 1 | NEXTSTEP task causes a new value to be loaded to internal compare registers |

## 6.12.5.18 LOOP

Address offset: 0x514

Number of playbacks of a loop

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | CNT | | | Number of playbacks of pattern cycles |
| | | | Disabled | 0 | Looping disabled (stop at the end of the sequence) |

## 6.12.5.19 SEQ[n].PTR (n=0..1)

Address offset: 0x520 + (n × 0x20)

Beginning address in RAM of this sequence

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | PTR | | | Beginning address in RAM of this sequence |

**Note:** See the memory chapter for details about which memories are available for EasyDMA.

## 6.12.5.20 SEQ[n].CNT (n=0..1)

Address offset: 0x524 + (n × 0x20)

Number of values (duty cycles) in this sequence

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | CNT | | | Number of values (duty cycles) in this sequence |
| | | | Disabled | 0 | Sequence is disabled, and shall not be started as it is empty |

### 6.12.5.21 SEQ[n].REFRESH (n=0..1)

Address offset: 0x528 + (n × 0x20)

Number of additional PWM periods between samples loaded into compare register

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000001** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | CNT | | | Number of additional PWM periods between samples loaded into compare register (load every REFRESH.CNT+1 PWM periods) |
| | | | Continuous | 0 | Update every PWM period |

### 6.12.5.22 SEQ[n].ENDDELAY (n=0..1)

Address offset: 0x52C + (n × 0x20)

Time added after the sequence

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | CNT | | | Time added after the sequence in PWM periods |

### 6.12.5.23 PSEL.OUT[n] (n=0..3)

Address offset: 0x560 + (n × 0x4)

Output pin select for PWM channel n

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.13 QDEC — Quadrature decoder

The Quadrature decoder (QDEC) provides buffered decoding of quadrature-encoded sensor signals. It is suitable for mechanical and optical sensors.

The sample period and accumulation are configurable to match application requirements. The QDEC provides the following:

- Decoding of digital waveform from off-chip quadrature encoder.
- Sample accumulation eliminating hard real-time requirements to be enforced on application.
- Optional input de-bounce filters.
- Optional LED output signal for optical encoders.

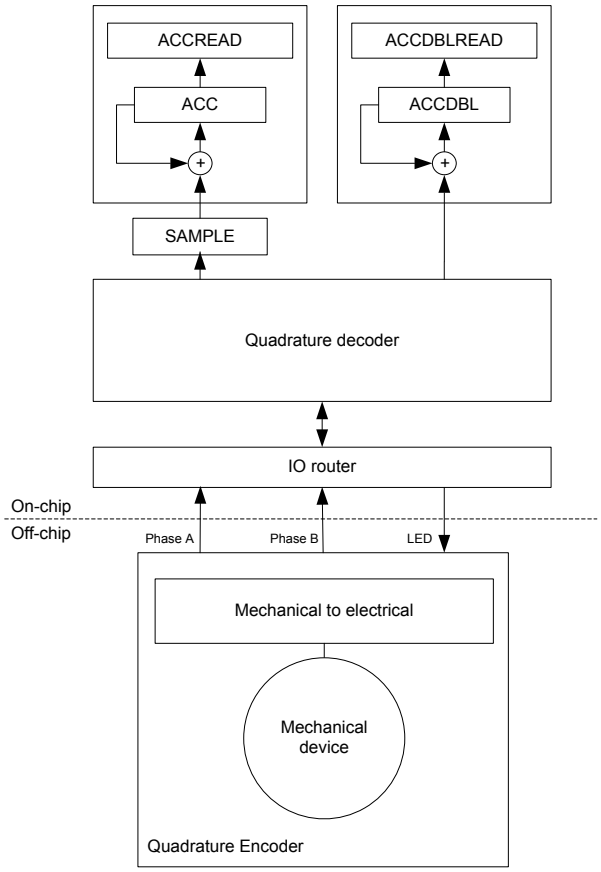



*Figure 56: Quadrature decoder configuration*

## 6.13.1 Sampling and decoding

The QDEC decodes the output from an incremental motion encoder by sampling the QDEC phase input pins (A and B).

The off-chip quadrature encoder is an incremental motion encoder outputting two waveforms, phase A and phase B. The two output waveforms are always 90 degrees out of phase, meaning that one always changes level before the other. The direction of movement is indicated by which of these two waveforms that changes level first. Invalid transitions may occur, that is when the two waveforms switch simultaneously. This may occur if the wheel rotates too fast relative to the sample rate set for the decoder.

The QDEC decodes the output from the off-chip encoder by sampling the QDEC phase input pins (A and B) at a fixed rate as specified in the SAMPLEPER register.

If the SAMPLEPER value needs to be changed, the QDEC shall be stopped using the STOP task. SAMPLEPER can be then changed upon receiving the STOPPED event, and QDEC can be restarted using the START task. Failing to do so may result in unpredictable behaviour.

It is good practice to change other registers (LEDPOL, REPORTPER, DBFEN and LEDPRE) only when the QDEC is stopped.

When started, the decoder continuously samples the two input waveforms and decodes these by comparing the current sample pair (n) with the previous sample pair (n-1).

The decoding of the sample pairs is described in the table below.

| Previous sample pair(n - 1) | | Current samples pair(n) | | SAMPLE register | ACC operation | ACCDBL operation | Description |
|---|---|---|---|---|---|---|---|
| A | B | A | B | | | | |
| 0 | 0 | 0 | 0 | 0 | No change | No change | No movement |
| 0 | 0 | 0 | 1 | 1 | Increment | No change | Movement in positive direction |
| 0 | 0 | 1 | 0 | -1 | Decrement | No change | Movement in negative direction |
| 0 | 0 | 1 | 1 | 2 | No change | Increment | Error: Double transition |
| 0 | 1 | 0 | 0 | -1 | Decrement | No change | Movement in negative direction |
| 0 | 1 | 0 | 1 | 0 | No change | No change | No movement |
| 0 | 1 | 1 | 0 | 2 | No change | Increment | Error: Double transition |
| 0 | 1 | 1 | 1 | 1 | Increment | No change | Movement in positive direction |
| 1 | 0 | 0 | 0 | 1 | Increment | No change | Movement in positive direction |
| 1 | 0 | 0 | 1 | 2 | No change | Increment | Error: Double transition |
| 1 | 0 | 1 | 0 | 0 | No change | No change | No movement |
| 1 | 0 | 1 | 1 | -1 | Decrement | No change | Movement in negative direction |
| 1 | 1 | 0 | 0 | 2 | No change | Increment | Error: Double transition |
| 1 | 1 | 0 | 1 | -1 | Decrement | No change | Movement in negative direction |
| 1 | 1 | 1 | 0 | 1 | Increment | No change | Movement in positive direction |
| 1 | 1 | 1 | 1 | 0 | No change | No change | No movement |

*Table 55: Sampled value encoding*

## 6.13.2 LED output

The LED output follows the sample period, and the LED is switched on a given period before sampling and switched off immediately after the inputs are sampled. The period the LED is switched on before sampling is given in the LEDPRE register.

The LED output pin polarity is specified in the LEDPOL register.

For using off-chip mechanical encoders not requiring a LED, the LED output can be disabled by writing value 'Disconnected' to the CONNECT field of the PSEL.LED register. In this case the QDEC will not acquire access to a LED output pin and the pin can be used for other purposes by the CPU.

## 6.13.3 Debounce filters

Each of the two-phase inputs have digital debounce filters.

When enabled through the DBFEN register, the filter inputs are sampled at a fixed 1 MHz frequency during the entire sample period (which is specified in the SAMPLEPER register), and the filters require all of the samples within this sample period to equal before the input signal is accepted and transferred to the output of the filter.

As a result, only input signal with a steady state longer than twice the period specified in SAMPLEPER are guaranteed to pass through the filter, and any signal with a steady state shorter than SAMPLEPER will always be suppressed by the filter. (This is assumed that the frequency during the debounce period never exceeds 500 kHz (as required by the Nyquist theorem when using a 1 MHz sample frequency).

The LED will always be ON when the debounce filters are enabled, as the inputs in this case will be sampled continuously.

Note that when when the debounce filters are enabled, displacements reported by the QDEC peripheral are delayed by one SAMPLEPER period.

## 6.13.4 Accumulators

The quadrature decoder contains two accumulator registers, ACC and ACCDBL, that accumulate respectively valid motion sample values and the number of detected invalid samples (double transitions).

The ACC register will accumulate all valid values (1/-1) written to the SAMPLE register. This can be useful for preventing hard real-time requirements from being enforced on the application. When using the ACC register the application does not need to read every single sample from the SAMPLE register, but can instead fetch the ACC register whenever it fits the application. The ACC register will always hold the relative movement of the external mechanical device since the previous clearing of the ACC register. Sample values indicating a double transition (2) will not be accumulated in the ACC register.

An ACCOF event will be generated if the ACC receives a SAMPLE value that would cause the register to overflow or underflow. Any SAMPLE value that would cause an ACC overflow or underflow will be discarded, but any samples not causing the ACC to overflow or underflow will still be accepted.

The accumulator ACCDBL accumulates the number of detected double transitions since the previous clearing of the ACCDBL register.

The ACC and ACCDBL registers can be cleared by the READCLRACC and subsequently read using the ACCREAD and ACCDBLREAD registers.

The ACC register can be separately cleared by the RDCLRACC and subsequently read using the ACCREAD registers.

The ACCDBL register can be separately cleared by the RDCLRDBL and subsequently read using the ACCDBLREAD registers.

The REPORTPER register allows automating the capture of several samples before it can send out a REPORTRDY event in case a non-null displacement has been captured and accumulated, and a DBLRDY event in case one or more double-displacements have been captured and accumulated. The REPORTPER field in this register selects after how many samples the accumulators contents are evaluated to send (or not) REPORTRDY and DBLRDY events.

Using the RDCLRACC task (manually sent upon receiving the event, or using the DBLRDY_RDCLRACC shortcut), ACCREAD can then be read.

In case at least one double transition has been captured and accumulated, a DBLRDY event is sent. Using the RDCLRDBL task (manually sent upon receiving the event, or using the DBLRDY_RDCLRDBL shortcut), ACCDBLREAD can then be read.

## 6.13.5 Output/input pins

The QDEC uses a three-pin interface to the off-chip quadrature encoder.

These pins will be acquired when the QDEC is enabled in the ENABLE register. The pins acquired by the QDEC cannot be written by the CPU, but they can still be read by the CPU.

The pin numbers to be used for the QDEC are selected using the PSEL.n registers.

## 6.13.6 Pin configuration

The Phase A, Phase B, and LED signals are mapped to physical pins according to the configuration specified in the PSEL.A, PSEL.B, and PSEL.LED registers respectively.

If the CONNECT field value 'Disconnected' is specified in any of these registers, the associated signal will not be connected to any physical pin. The PSEL.A, PSEL.B, and PSEL.LED registers and their configurations are only used as long as the QDEC is enabled, and retained only as long as the device is in ON mode.

When the peripheral is disabled, the pins will behave as regular GPIOs, and use the configuration in their respective OUT bit field and PIN_CNF[n] register.

To secure correct behavior in the QDEC, the pins used by the QDEC must be configured in the GPIO peripheral as described in GPIO configuration before enabling peripheral on page 169 before enabling the QDEC. This configuration must be retained in the GPIO for the selected IOs as long as the QDEC is enabled.

Only one peripheral can be assigned to drive a particular GPIO pin at a time. Failing to do so may result in unpredictable behavior.

| QDEC signal | QDEC pin | Direction | Output value | Comment |
|---|---|---|---|---|
| Phase A | As specified in PSEL.A | Input | Not applicable | |
| Phase B | As specified in PSEL.B | Input | Not applicable | |
| LED | As specified in PSEL.LED | Input | Not applicable | |

*Table 56: GPIO configuration before enabling peripheral*

## 6.13.7 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40012000 | QDEC | QDEC | Quadrature decoder | |

*Table 57: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_START | 0x000 | Task starting the quadrature decoder |
| TASKS_STOP | 0x004 | Task stopping the quadrature decoder |
| TASKS_READCLRACC | 0x008 | Read and clear ACC and ACCDBL |
| TASKS_RDCLRACC | 0x00C | Read and clear ACC |
| TASKS_RDCLRDBL | 0x010 | Read and clear ACCDBL |
| EVENTS_SAMPLERDY | 0x100 | Event being generated for every new sample value written to the SAMPLE register |
| EVENTS_REPORTRDY | 0x104 | Non-null report ready |
| EVENTS_ACCOF | 0x108 | ACC or ACCDBL register overflow |
| EVENTS_DBLRDY | 0x10C | Double displacement(s) detected |
| EVENTS_STOPPED | 0x110 | QDEC has been stopped |
| SHORTS | 0x200 | Shortcuts between local events and tasks |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| ENABLE | 0x500 | Enable the quadrature decoder |
| LEDPOL | 0x504 | LED output pin polarity |
| SAMPLEPER | 0x508 | Sample period |
| SAMPLE | 0x50C | Motion sample value |
| REPORTPER | 0x510 | Number of samples to be taken before REPORTRDY and DBLRDY events can be generated |
| ACC | 0x514 | Register accumulating the valid transitions |
| ACCREAD | 0x518 | Snapshot of the ACC register, updated by the READCLRACC or RDCLRACC task |
| PSEL.LED | 0x51C | Pin select for LED signal |
| PSEL.A | 0x520 | Pin select for A signal |
| PSEL.B | 0x524 | Pin select for B signal |
| DBFEN | 0x528 | Enable input debounce filters |
| LEDPRE | 0x540 | Time period the LED is switched ON prior to sampling |
| ACCDBL | 0x544 | Register accumulating the number of detected double transitions |

| Register | Offset | Description |
|---|---|---|
| ACCDBLREAD | 0x548 | Snapshot of the ACCDBL, updated by the READCLRACC or RDCLRDBL task |

*Table 58: Register overview*

## 6.13.7.1 TASKS_START

Address offset: 0x000

Task starting the quadrature decoder

When started, the SAMPLE register will be continuously updated at the rate given in the SAMPLEPER register.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_START | | | Task starting the quadrature decoder<br>When started, the SAMPLE register will be continuously updated at the rate given in the SAMPLEPER register. |
| | | | Trigger | 1 | Trigger task |

## 6.13.7.2 TASKS_STOP

Address offset: 0x004

Task stopping the quadrature decoder

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STOP | | | Task stopping the quadrature decoder |
| | | | Trigger | 1 | Trigger task |

## 6.13.7.3 TASKS_READCLRACC

Address offset: 0x008

Read and clear ACC and ACCDBL

Task transferring the content of ACC to ACCREAD and the content of ACCDBL to ACCDBLREAD, and then clearing the ACC and ACCDBL registers. These read-and-clear operations will be done atomically.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_READCLRACC | | | Read and clear ACC and ACCDBL<br>Task transferring the content of ACC to ACCREAD and the content of ACCDBL to ACCDBLREAD, and then clearing the ACC and ACCDBL registers. These read-and-clear operations will be done atomically. |
| | | | Trigger | 1 | Trigger task |

### 6.13.7.4 TASKS_RDCLRACC

Address offset: 0x00C

Read and clear ACC

Task transferring the content of ACC to ACCREAD, and then clearing the ACC register. This read-and-clear operation will be done atomically.

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_RDCLRACC | | | Read and clear ACC<br>Task transferring the content of ACC to ACCREAD, and then clearing the ACC register. This read-and-clear operation will be done atomically. |
| | | | Trigger | 1 | Trigger task |

### 6.13.7.5 TASKS_RDCLRDBL

Address offset: 0x010

Read and clear ACCDBL

Task transferring the content of ACCDBL to ACCDBLREAD, and then clearing the ACCDBL register. This read-and-clear operation will be done atomically.

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_RDCLRDBL | | | Read and clear ACCDBL<br>Task transferring the content of ACCDBL to ACCDBLREAD, and then clearing the ACCDBL register. This read-and-clear operation will be done atomically. |
| | | | Trigger | 1 | Trigger task |

### 6.13.7.6 EVENTS_SAMPLERDY

Address offset: 0x100

Event being generated for every new sample value written to the SAMPLE register

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_SAMPLERDY | | | Event being generated for every new sample value written to the SAMPLE register |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.13.7.7 EVENTS_REPORTRDY

Address offset: 0x104

Non-null report ready

Event generated when REPORTPER number of samples has been accumulated in the ACC register and the content of the ACC register is not equal to 0. (Thus, this event is only generated if a motion is detected since the previous clearing of the ACC register).

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_REPORTRDY | | | Non-null report ready<br>Event generated when REPORTPER number of samples has been accumulated in the ACC register and the content of the ACC register is not equal to 0. (Thus, this event is only generated if a motion is detected since the previous clearing of the ACC register). |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.13.7.8 EVENTS_ACCOF

Address offset: 0x108

ACC or ACCDBL register overflow

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_ACCOF | | | ACC or ACCDBL register overflow |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.13.7.9 EVENTS_DBLRDY

Address offset: 0x10C

Double displacement(s) detected

Event generated when REPORTPER number of samples has been accumulated and the content of the ACCDBL register is not equal to 0. (Thus, this event is only generated if a double transition is detected since the previous clearing of the ACCDBL register).

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | EVENTS_DBLRDY | | | Double displacement(s) detected<br>Event generated when REPORTPER number of samples has been accumulated and the content of the ACCDBL register is not equal to 0. (Thus, this event is only generated if a double transition is detected since the previous clearing of the ACCDBL register). |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.13.7.10 EVENTS_STOPPED

Address offset: 0x110

QDEC has been stopped

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | EVENTS_STOPPED | | | QDEC has been stopped |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.13.7.11 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | REPORTRDY_READCLRACC | | | Shortcut between event REPORTRDY and task READCLRACC |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| B | RW | SAMPLERDY_STOP | | | Shortcut between event SAMPLERDY and task STOP |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| C | RW | REPORTRDY_RDCLRACC | | | Shortcut between event REPORTRDY and task RDCLRACC |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| D | RW | REPORTRDY_STOP | | | Shortcut between event REPORTRDY and task STOP |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| E | RW | DBLRDY_RDCLRDBL | | | Shortcut between event DBLRDY and task RDCLRDBL |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| F | RW | DBLRDY_STOP | | | Shortcut between event DBLRDY and task STOP |
| | | | Disabled | 0 | Disable shortcut |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| | | | Enabled | 1 | Enable shortcut |
| G | RW | SAMPLERDY_READCLRACC | | | Shortcut between event SAMPLERDY and task READCLRACC |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |

## 6.13.7.12 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | SAMPLERDY | | | Write '1' to enable interrupt for event SAMPLERDY |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | REPORTRDY | | | Write '1' to enable interrupt for event REPORTRDY<br>Event generated when REPORTPER number of samples has been accumulated in the ACC register and the content of the ACC register is not equal to 0. (Thus, this event is only generated if a motion is detected since the previous clearing of the ACC register). |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | ACCOF | | | Write '1' to enable interrupt for event ACCOF |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | DBLRDY | | | Write '1' to enable interrupt for event DBLRDY<br>Event generated when REPORTPER number of samples has been accumulated and the content of the ACCDBL register is not equal to 0. (Thus, this event is only generated if a double transition is detected since the previous clearing of the ACCDBL register). |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E | RW | STOPPED | | | Write '1' to enable interrupt for event STOPPED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.13.7.13 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | SAMPLERDY | | | Write '1' to disable interrupt for event SAMPLERDY |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | REPORTRDY | | | Write '1' to disable interrupt for event REPORTRDY<br>Event generated when REPORTPER number of samples has been accumulated in the ACC register and the content of the ACC register is not equal to 0. (Thus, this event is only generated if a motion is detected since the previous clearing of the ACC register). |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | ACCOF | | | Write '1' to disable interrupt for event ACCOF |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | DBLRDY | | | Write '1' to disable interrupt for event DBLRDY<br>Event generated when REPORTPER number of samples has been accumulated and the content of the ACCDBL register is not equal to 0. (Thus, this event is only generated if a double transition is detected since the previous clearing of the ACCDBL register). |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E | RW | STOPPED | | | Write '1' to disable interrupt for event STOPPED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.13.7.14 ENABLE

Address offset: 0x500

Enable the quadrature decoder

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ENABLE | | | Enable or disable the quadrature decoder<br>When enabled the decoder pins will be active. When disabled the quadrature decoder pins are not active and can be used as GPIO . |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |

## 6.13.7.15 LEDPOL

Address offset: 0x504

LED output pin polarity

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | LEDPOL | | | LED output pin polarity |
| | | | ActiveLow | 0 | Led active on output pin low |
| | | | ActiveHigh | 1 | Led active on output pin high |

## 6.13.7.16 SAMPLEPER

Address offset: 0x508

Sample period

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | SAMPLEPER | | | Sample period. The SAMPLE register will be updated for every new sample |
| | | | 128us | 0 | 128 us |
| | | | 256us | 1 | 256 us |
| | | | 512us | 2 | 512 us |
| | | | 1024us | 3 | 1024 us |
| | | | 2048us | 4 | 2048 us |
| | | | 4096us | 5 | 4096 us |
| | | | 8192us | 6 | 8192 us |
| | | | 16384us | 7 | 16384 us |
| | | | 32ms | 8 | 32768 us |
| | | | 65ms | 9 | 65536 us |
| | | | 131ms | 10 | 131072 us |

## 6.13.7.17 SAMPLE

Address offset: 0x50C

Motion sample value

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | SAMPLE | | [-1..2] | Last motion sample<br>The value is a 2's complement value, and the sign gives the direction of the motion. The value '2' indicates a double transition. |

## 6.13.7.18 REPORTPER

Address offset: 0x510

Number of samples to be taken before REPORTRDY and DBLRDY events can be generated

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | REPORTPER | | | Specifies the number of samples to be accumulated in the ACC register before the REPORTRDY and DBLRDY events can be generated<br><br>The report period in [us] is given as: RPUS = SP * RP Where RPUS is the report period in [us/report], SP is the sample period in [us/sample] specified in SAMPLEPER, and RP is the report period in [samples/report] specified in REPORTPER . |
| | | | 10Smpl | 0 | 10 samples / report |
| | | | 40Smpl | 1 | 40 samples / report |
| | | | 80Smpl | 2 | 80 samples / report |
| | | | 120Smpl | 3 | 120 samples / report |
| | | | 160Smpl | 4 | 160 samples / report |
| | | | 200Smpl | 5 | 200 samples / report |
| | | | 240Smpl | 6 | 240 samples / report |
| | | | 280Smpl | 7 | 280 samples / report |
| | | | 1Smpl | 8 | 1 sample / report |

## 6.13.7.19 ACC

Address offset: 0x514

Register accumulating the valid transitions

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | ACC | | [-1024..1023] | Register accumulating all valid samples (not double transition) read from the SAMPLE register<br><br>Double transitions ( SAMPLE = 2 ) will not be accumulated in this register. The value is a 32 bit 2's complement value. If a sample that would cause this register to overflow or underflow is received, the sample will be ignored and an overflow event ( ACCOF ) will be generated. The ACC register is cleared by triggering the READCLRACC or the RDCLRACC task. |

## 6.13.7.20 ACCREAD

Address offset: 0x518

Snapshot of the ACC register, updated by the READCLRACC or RDCLRACC task

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | ACCREAD | | [-1024..1023] | Snapshot of the ACC register.<br>The ACCREAD register is updated when the READCLRACC or RDCLRACC task is triggered |

## 6.13.7.21 PSEL.LED

Address offset: 0x51C

Pin select for LED signal

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.13.7.22 PSEL.A

Address offset: 0x520

Pin select for A signal

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.13.7.23 PSEL.B

Address offset: 0x524

Pin select for B signal

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.13.7.24 DBFEN

Address offset: 0x528

Enable input debounce filters

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | DBFEN | | | Enable input debounce filters |
| | | | Disabled | 0 | Debounce input filters disabled |
| | | | Enabled | 1 | Debounce input filters enabled |

## 6.13.7.25 LEDPRE

Address offset: 0x540

Time period the LED is switched ON prior to sampling

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A |
| **Reset 0x00000010** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | LEDPRE | | [1..511] | Period in us the LED is switched on prior to sampling |

## 6.13.7.26 ACCDBL

Address offset: 0x544

Register accumulating the number of detected double transitions

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | ACCDBL | | [0..15] | Register accumulating the number of detected double or illegal transitions. ( SAMPLE = 2 ).<br><br>When this register has reached its maximum value the accumulation of double / illegal transitions will stop. An overflow event ( ACCOF ) will be generated if any double or illegal transitions are detected after the maximum value was reached. This field is cleared by triggering the READCLRACC or RDCLRDBL task. |

## 6.13.7.27 ACCDBLREAD

Address offset: 0x548

Snapshot of the ACCDBL, updated by the READCLRACC or RDCLRDBL task

| Bit number | 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 |
|---|---|
| ID | A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | ACCDBLREAD | | [0..15] | Snapshot of the ACCDBL register. This field is updated when the READCLRACC or RDCLRDBL task is triggered. |

## 6.13.8 Electrical specification

### 6.13.8.1 QDEC Electrical Specification

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{SAMPLE}$ | Time between sampling signals from quadrature decoder | 128 | | 131072 | µs |
| $t_{LED}$ | Time from LED is turned on to signals are sampled | 0 | | 511 | µs |

# 6.14 RADIO — 2.4 GHz radio

The 2.4 GHz radio transceiver is compatible with multiple radio standards such as 1 Mbps, 2 Mbps and long range *Bluetooth*® low energy. IEEE 802.15.4 250 kbps mode is fully supported as well as Nordic's proprietary 1 Mbps and 2 Mbps modes of operation.

Listed here are main features for the RADIO:

- Multidomain 2.4 GHz radio transceiver
  - 1 Mbps, 2 Mbps, and long range (125 kbps and 500 kbps mode) *Bluetooth*® low energy modes
  - Angle-of-arrival (AoA) and angle-of-departure (AoD) direction finding using *Bluetooth*® low energy
  - 250 kbps IEEE 802.15.4 mode
  - 1 Mbps and 2 Mbps Nordic proprietary modes
- Best in class link budget and low power operation
- Efficient data interface with EasyDMA support
- Automatic address filtering and pattern matching

EasyDMA, in combination with an automated packet assembler, packet disassembler, automated CRC generator and CRC checker, makes it easy to configure and use the RADIO. See the following figure for details.

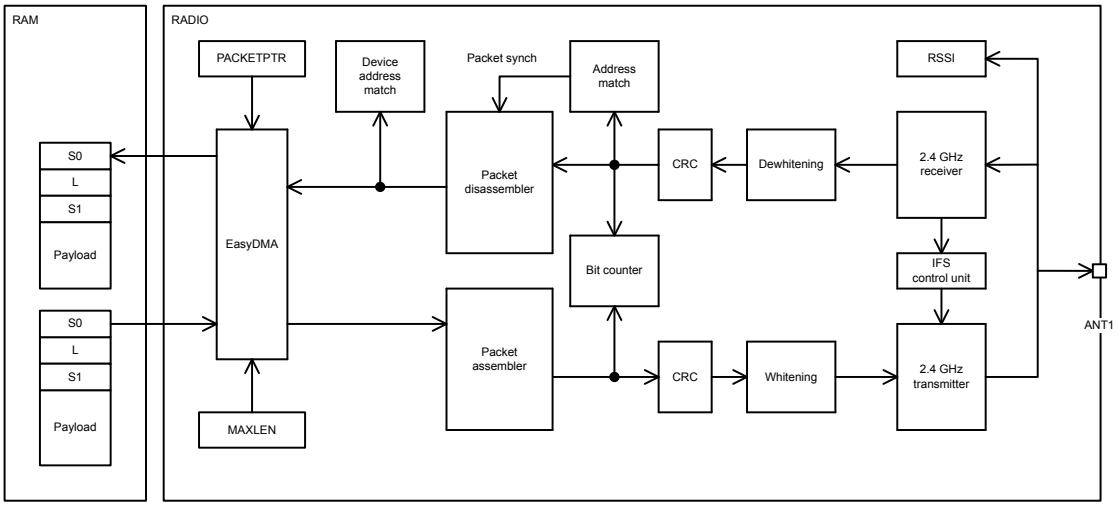



*Figure 57: RADIO block diagram*

The RADIO includes a device address match unit and an interframe spacing control unit that can be utilized to simplify address whitelisting and interframe spacing respectively in *Bluetooth*® low energy and similar applications.

The RADIO also includes a received signal strength indicator (RSSI) and a bit counter. The bit counter generates events when a preconfigured number of bits are sent or received by the RADIO.

## 6.14.1 Packet configuration

A RADIO packet contains the following fields: PREAMBLE, ADDRESS, S0, LENGTH, S1, PAYLOAD, and CRC.

The content of a RADIO packet is illustrated in the figure below. The RADIO sends the fields in the packet according to the order illustrated in the following figure, starting on the left.

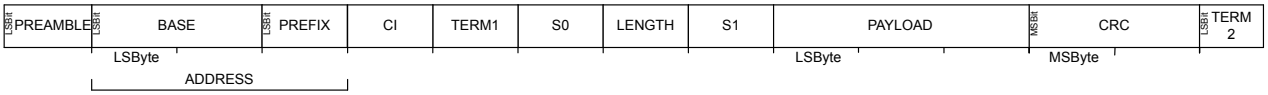



*Figure 58: On-air packet layout*

Not shown in the figure is the static payload add-on (the length of which is defined in PCNF1.STATLEN, and which is 0 bytes in a standard BLE packet). The static payload add-on is sent between PAYLOAD and CRC fields. The RADIO sends the different fields in the packet in the order they are illustrated above, from left to right. The preamble will be sent with least significant bit first on air.

PREAMBLE is sent with least significant bit first on air. The size of the PREAMBLE depends on the mode selected in the MODE register:

- The PREAMBLE is one byte for MODE = Ble_1Mbit as well as all Nordic proprietary operating modes (MODE = Nrf_1Mbit and MODE = Nrf_2Mbit), and PCNF0.PLEN has to be set accordingly. If the first bit of the ADDRESS is 0, the preamble will be set to 0xAA. Otherwise the PREAMBLE will be set to 0x55.
- For MODE = Ble_2Mbit, the PREAMBLE must be set to 2 byte through PCNF0.PLEN. If the first bit of the ADDRESS is 0, the preamble will be set to 0xAAAA. Otherwise the PREAMBLE will be set to 0x5555.
- For MODE = Ble_LR125Kbit and MODE = Ble_LR500Kbit, the PREAMBLE is 10 repetitions of 0x3C.
- For MODE = Ieee802154_250Kbit, the PREAMBLE is 4 bytes and set to all zeros.

Radio packets are stored in memory inside instances of a RADIO packet data structure as illustrated below. The PREAMBLE, ADDRESS, CI, TERM1, TERM2, and CRC fields are omitted in this data structure. Fields S0, LENGTH, and S1 are optional.

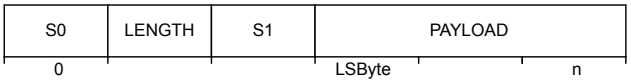



*Figure 59: In-RAM representation of RADIO packet*

The byte ordering on air is always least significant byte first for the ADDRESS and PAYLOAD fields, and most significant byte first for the CRC field. The ADDRESS fields are always transmitted and received least significant bit first. The CRC field is always transmitted and received most significant bit first. The bit-endian, i.e. the order in which the bits are sent and received, of the S0, LENGTH, S1, and PAYLOAD fields can be configured via PCNF1.ENDIAN.

The sizes of the S0, LENGTH and S1 fields can be individually configured via S0LEN, LFLEN, and S1LEN in PCNF0 respectively. If any of these fields are configured to be less than 8 bits, the least significant bits of the fields are used.

If S0, LENGTH, or S1 are specified with zero length, their fields will be omitted in memory. Otherwise each field will be represented as a separate byte, regardless of the number of bits in their on-air counterpart.

Independent of the configuration of PCNF1.MAXLEN, the combined length of S0, LENGTH, S1, and PAYLOAD cannot exceed 258 bytes.

## 6.14.2 Address configuration

The on-air radio ADDRESS field is composed of two parts, the base address field and the address prefix field.

The size of the base address field is configurable via PCNF1.BALEN. The base address is truncated from the least significant byte if the PCNF1.BALEN is less than 4. See Definition of logical addresses on page 182.

| Logical address | Base address | Prefix byte |
|---|---|---|
| 0 | BASE0 | PREFIX0.AP0 |
| 1 | BASE1 | PREFIX0.AP1 |
| 2 | BASE1 | PREFIX0.AP2 |
| 3 | BASE1 | PREFIX0.AP3 |
| 4 | BASE1 | PREFIX1.AP4 |
| 5 | BASE1 | PREFIX1.AP5 |
| 6 | BASE1 | PREFIX1.AP6 |
| 7 | BASE1 | PREFIX1.AP7 |

*Table 59: Definition of logical addresses*

The on-air addresses are defined in the BASE0/BASE1 and PREFIX0/PREFIX1 registers. It is only when writing these registers that the user must relate to the actual on-air addresses. For other radio address registers, such as the TXADDRESS, RXADDRESSES, and RXMATCH registers, logical radio addresses ranging from 0 to 7 are being used. The relationship between the on-air radio addresses and the logical addresses is described in Definition of logical addresses on page 182.

## 6.14.3 Data whitening

The RADIO is able to do packet whitening and de-whitening, enabled in PCNF1.WHITEEN. When enabled, whitening and de-whitening will be handled by the RADIO automatically as packets are sent and received.

The whitening word is generated using polynomial $g(D) = D^7 + D^4 + 1$, which then is XORed with the data packet that is to be whitened, or de-whitened. See the figure below.

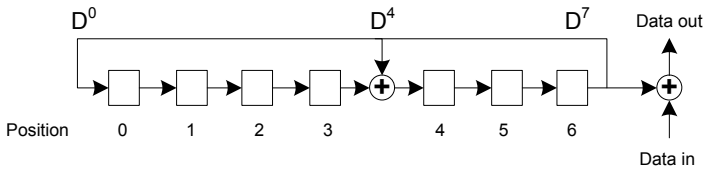



*Figure 60: Data whitening and de-whitening*

Whitening and de-whitening will be performed over the whole packet (except for the preamble and the address field).

The linear feedback shift register in the figure above is initialized via register DATAWHITEIV on page 228.

## 6.14.4 CRC

The CRC generator in the RADIO calculates the CRC over the whole packet excluding the preamble. If desirable, the address field can be excluded from the CRC calculation as well.

See CRCCNF register for more information.

The CRC polynomial is configurable as illustrated in the following figure, where bit 0 in the CRCPOLY register corresponds to $X^0$ and bit 1 corresponds to $X^1$ etc. See CRCPOLY on page 227 for more information.

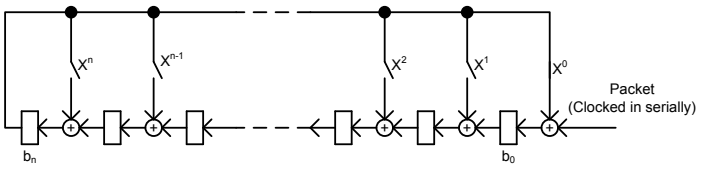



*Figure 61: CRC generation of an n bit CRC*

The figure shows that the CRC is calculated by feeding the packet serially through the CRC generator. Before the packet is clocked through the CRC generator, the CRC generator's latches $b_0$ through $b_n$ will be initialized with a predefined value specified in the CRCINIT register. After the whole packet has been clocked through the CRC generator, $b_0$ through $b_n$ will hold the resulting CRC. This value will be used by the RADIO during both transmission and reception. Latches $b_0$ through $b_n$ are not available to be read by the CPU at any time. However, a received CRC can be read by the CPU via the RXCRC register.

The length (n) of the CRC is configurable, see CRCCNF for more information.

Once the entire packet, including the CRC, has been received and no errors were detected, the RADIO generates a CRCOK event. If CRC errors were detected, a CRCERROR event is generated.

The status of the CRC check can be read from the CRCSTATUS register after a packet has been received.

## 6.14.5 Radio states

Tasks and events are used to control the operating state of the RADIO.

The RADIO can enter the states described the table below.

| State | Description |
|---|---|
| DISABLED | No operations are going on inside the RADIO and the power consumption is at a minimum |
| RXRU | The RADIO is ramping up and preparing for reception |
| RXIDLE | The RADIO is ready for reception to start |
| RX | Reception has been started and the addresses enabled in the RXADDRESSES register are being monitored |
| TXRU | The RADIO is ramping up and preparing for transmission |
| TXIDLE | The RADIO is ready for transmission to start |
| TX | The RADIO is transmitting a packet |
| RXDISABLE | The RADIO is disabling the receiver |
| TXDISABLE | The RADIO is disabling the transmitter |

*Table 60: RADIO state diagram*

A state diagram showing an overview of the RADIO is shown in the following figure.

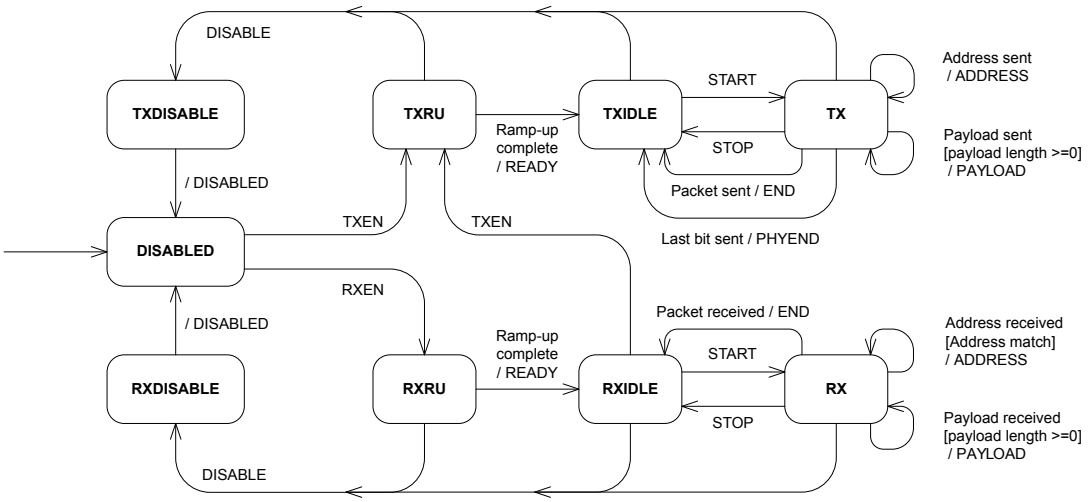



*Figure 62: Radio states*

This figure shows how the tasks and events relate to the RADIO's operation. The RADIO does not prevent a task from being triggered from the wrong state. If a task is triggered from the wrong state, for example if the RXEN task is triggered from the RXDISABLE state, this may lead to incorrect behavior. The PAYLOAD event is always generated even if the payload is zero.

The END to START shortcut should not be used with Ble_LR125Kbit, Ble_LR500Kbit, and Ieee802154_250Kbit modes. Rather the PHYEND to START shortcut.

## 6.14.6 Transmit sequence

Before the RADIO is able to transmit a packet, it must first ramp-up in TX mode. See TXRU in Radio states on page 184 and Transmit sequence on page 185. A TXRU ramp-up sequence is initiated when the TXEN task is triggered. After the RADIO has successfully ramped up it will generate the READY event indicating that a packet transmission can be initiated. A packet transmission is initiated by triggering the START task. The START task can first be triggered after the RADIO has entered into the TXIDLE state.

The following figure illustrates a single packet transmission where the CPU manually triggers the different tasks needed to control the flow of the RADIO, i.e. no shortcuts are used. If shortcuts are not used, a certain amount of delay caused by CPU execution is expected between READY and START, and between END and DISABLE. As illustrated in Transmit sequence on page 185 the RADIO will by default transmit '1's between READY and START, and between END and DISABLED. What is transmitted can be programmed through the DTX field in the MODECNF0 register.

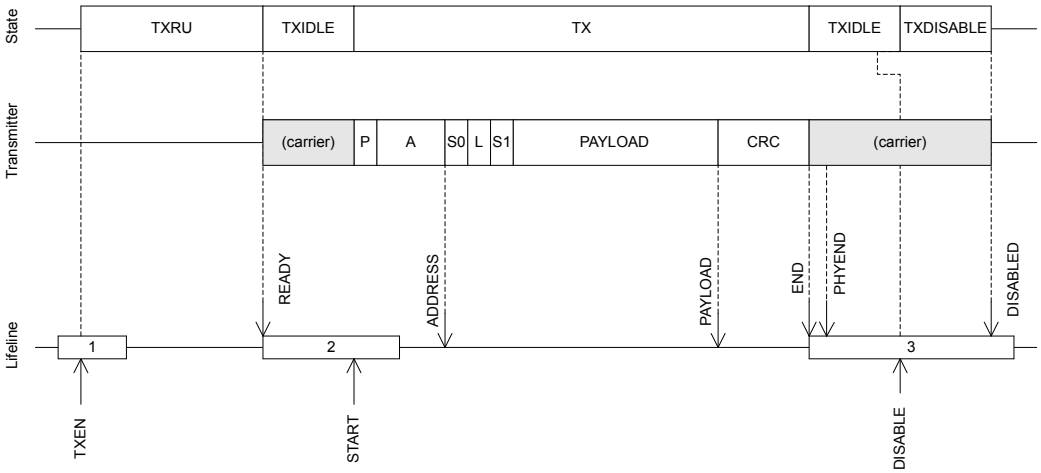



*Figure 63: Transmit sequence*

The following figure shows a slightly modified version of the transmit sequence where the RADIO is configured to use shortcuts between READY and START, and between END and DISABLE, which means that no delay is introduced.

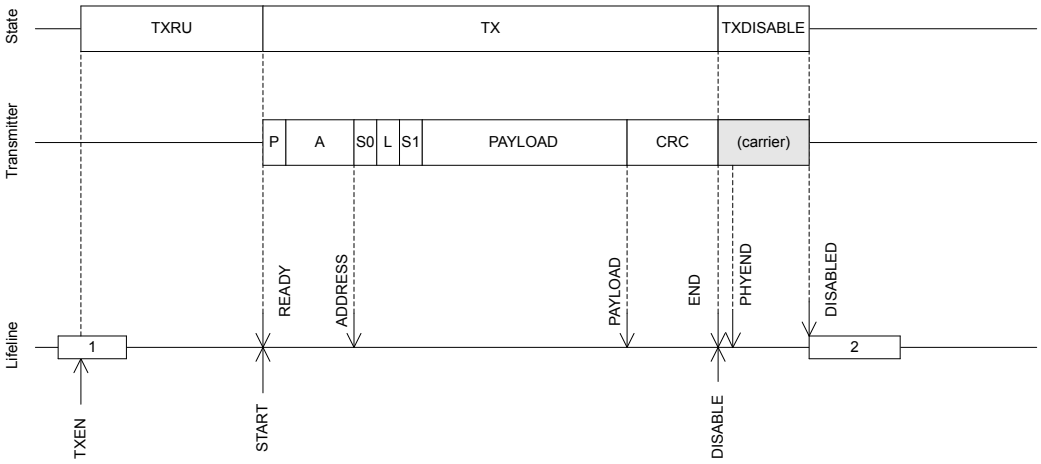



*Figure 64: Transmit sequence using shortcuts to avoid delays*

The RADIO is able to send multiple packets one after the other without having to disable and re-enable the RADIO between packets, as illustrated in the following figure.

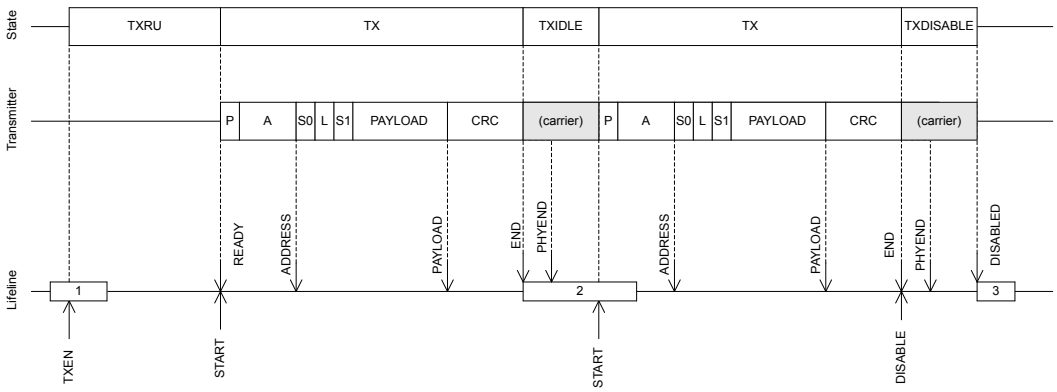



*Figure 65: Transmission of multiple packets*

## 6.14.7 Receive sequence

Before the RADIO is able to receive a packet, it must first ramp up in RX mode, see RXRU in Radio states on page 184 and Receive sequence on page 186.

An RXRU ramp up sequence is initiated when the RXEN task is triggered. After the RADIO has successfully ramped up it will generate the READY event indicating that a packet reception can be initiated. A packet reception is initiated by triggering the START task. As illustrated in Radio states on page 184, the START task can first be triggered after the RADIO has entered into the RXIDLE state.

The following figure shows a single packet reception where the CPU manually triggers the different tasks needed to control the flow of the RADIO, i.e. no shortcuts are used. If shortcuts are not used, a certain amount of delay caused by CPU execution is expected between READY and START, and between END and DISABLE. The RADIO will be listening and possibly receiving undefined data, represented with an 'X', from START and until a packet with valid preamble (P) is received.

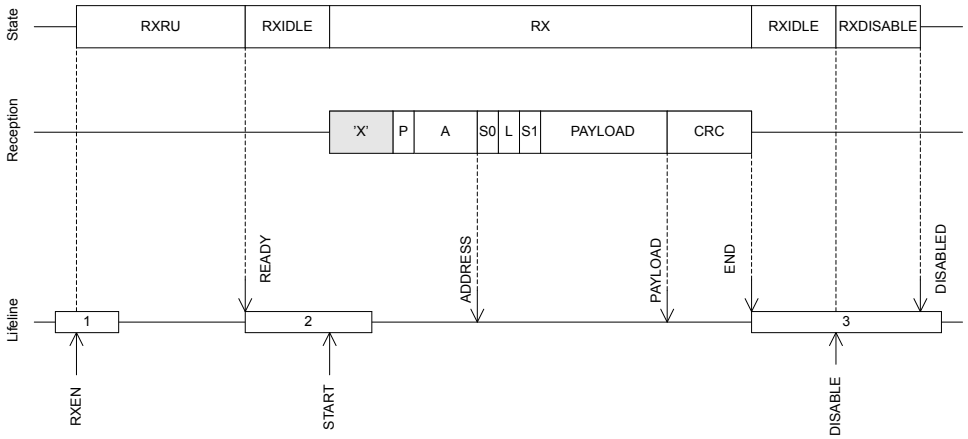



*Figure 66: Receive sequence*

The following figure shows a slightly modified version of the receive sequence, where the RADIO is configured to use shortcuts between READY and START, and between END and DISABLE, which means that no delay is introduced.

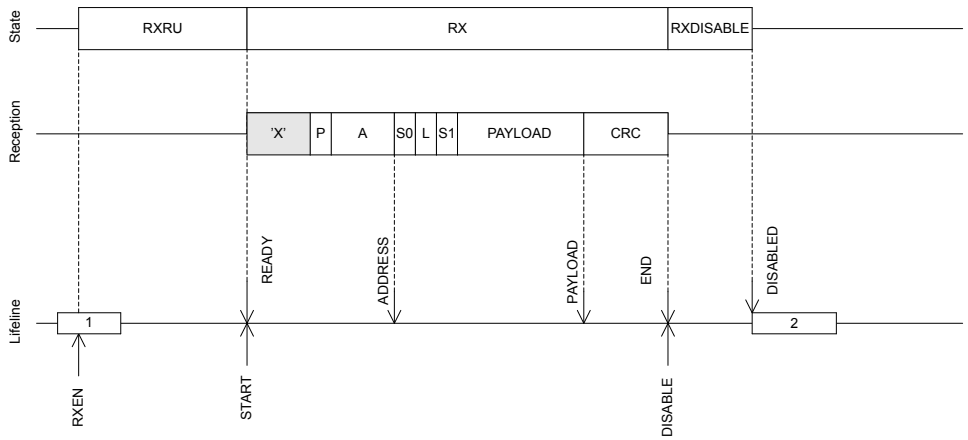



*Figure 67: Receive sequence using shortcuts to avoid delays*

The RADIO is able to receive consecutive packets without having to disable and re-enable the RADIO between packets, as illustrated in the figure below.

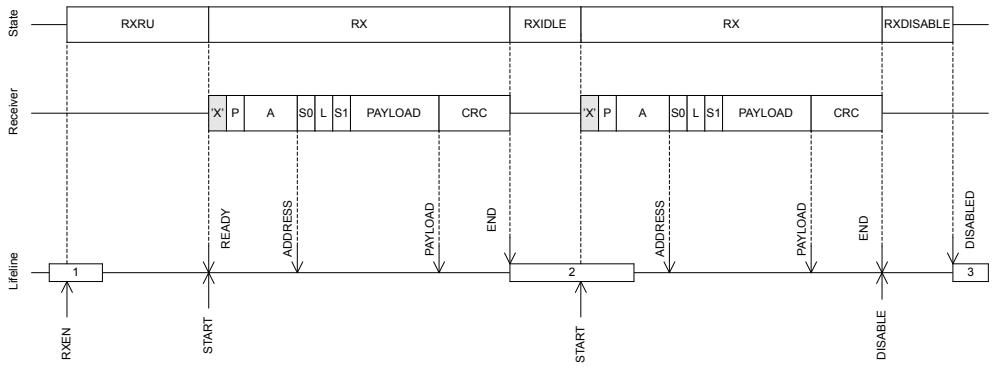



*Figure 68: Reception of multiple packets*

## 6.14.8 Received signal strength indicator (RSSI)

The RADIO implements a mechanism for measuring the power in the received signal. This feature is called received signal strength indicator (RSSI).

The RSSI is measured continuously and the value filtered using a single-pole IIR filter. After a signal level change, the RSSI will settle after approximately $RSSI_{SETTLE}$.

Sampling of the received signal strength is started by using the RSSISTART task. The sample can be read from the RSSISAMPLE register.

The sample period of the RSSI is defined by $RSSI_{PERIOD}$. The RSSISAMPLE will hold the pre-filtered received signal strength after this sample period.

For the RSSI sample to be valid, the RADIO has to be enabled in receive mode (RXEN task) and the reception has to be started (READY event followed by START task).

## 6.14.9 Interframe spacing (IFS)

Interframe spacing (IFS) is defined as the time, in microseconds, between two consecutive packets, starting from when the end of the last bit of the previous packet is received, to the beginning of the first bit of the subsequent packet that is transmitted. The RADIO is able to enforce this interval, as specified in the TIFS register, as long as the TIFS is not specified to be shorter than the RADIO's turnaround time, i.e. the time needed to switch off the receiver, and then switch the transmitter back on. The TIFS register can be written any time before the last bit on air is received.

This timing is illustrated in the figure below.

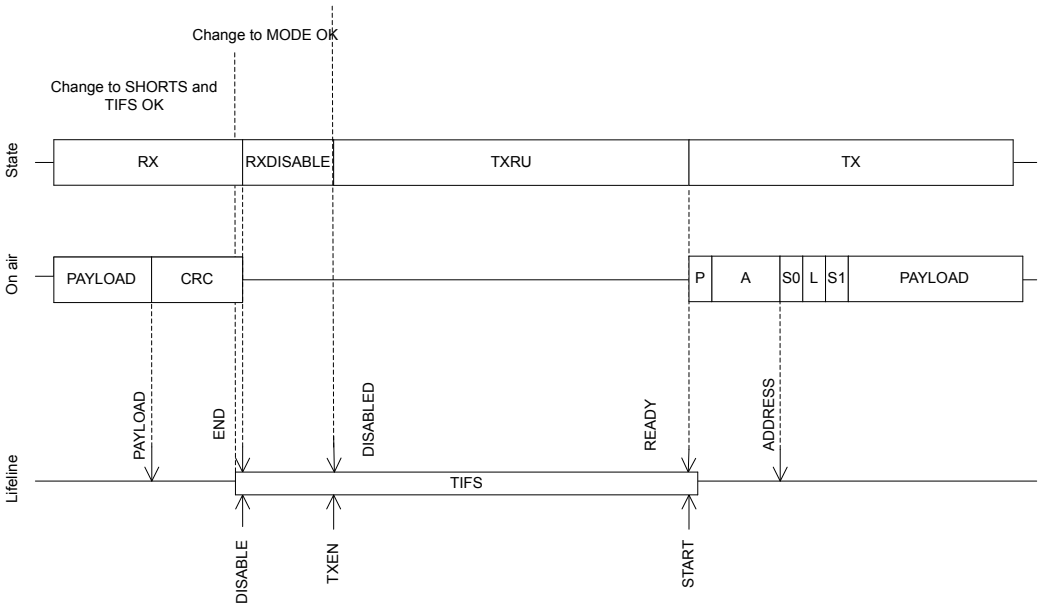



*Figure 69: IFS timing detail*

The TIFS duration starts after the last bit on air (just before the END event), and elapses with first bit being transmitted on air (just after READY event).

TIFS is only enforced if the shortcuts END-to-DISABLE and DISABLED-to-TXEN or END-to-DISABLE and DISABLED-to-RXEM are enabled. TIFS is qualified for use when MODE is either Ble_1Mbit, Ble_2Mbit, Ble_LR125Kbit, Ble_LR500Kbit, or Ieee802154_250Kbit, using the default ramp-up mode. SHORTS and TIFS are not double-buffered, and can be updated at any point in time before the last bit on air is received. The MODE register is double-buffered and sampled at the TXEN or RXEN task.

## 6.14.10 Device address match

The device address match feature is tailored for address whitelisting in *Bluetooth*® low energy and similar implementations.

This feature enables on-the-fly device address matching while receiving a packet on air. This feature only works in receive mode and when the RADIO is configured for little endian, see PCNF1.ENDIAN.

The device address match unit assumes that the first 48 bits of the payload are the device address and that bit number 6 in S0 is the TxAdd bit. See the *Bluetooth*® Core Specification for more information about device addresses, TxAdd, and whitelisting.

The RADIO is able to listen for eight different device addresses at the same time. These addresses are specified in a DAB/DAP register pair, one pair per address, in addition to a TxAdd bit configured in the DACNF register. The DAB register specifies the 32 least significant bits of the device address, while the DAP register specifies the 16 most significant bits of the device address.

Each of the device addresses can be individually included or excluded from the matching mechanism. This is configured in the DACNF register.

## 6.14.11 Bit counter

The RADIO implements a simple counter that can be configured to generate an event after a specific number of bits have been transmitted or received.

By using shortcuts, this counter can be started from different events generated by the RADIO and count relative to these.

The bit counter is started by triggering the BCSTART task, and stopped by triggering the BCSTOP task. A BCMATCH event will be generated when the bit counter has counted the number of bits specified in the

BCC register. The bit counter will continue to count bits until the DISABLED event is generated or until the BCSTOP task is triggered. The CPU can therefore, after a BCMATCH event, reconfigure the BCC value for new BCMATCH events within the same packet.

The bit counter can only be started after the RADIO has received the ADDRESS event.

The bit counter will stop and reset on either the BCSTOP, STOP, or DISABLE task, or the END event.

The figure below illustrates how the bit counter can be used to generate a BCMATCH event in the beginning of the packet payload, and again generate a second BCMATCH event after sending 2 bytes (16 bits) of the payload.

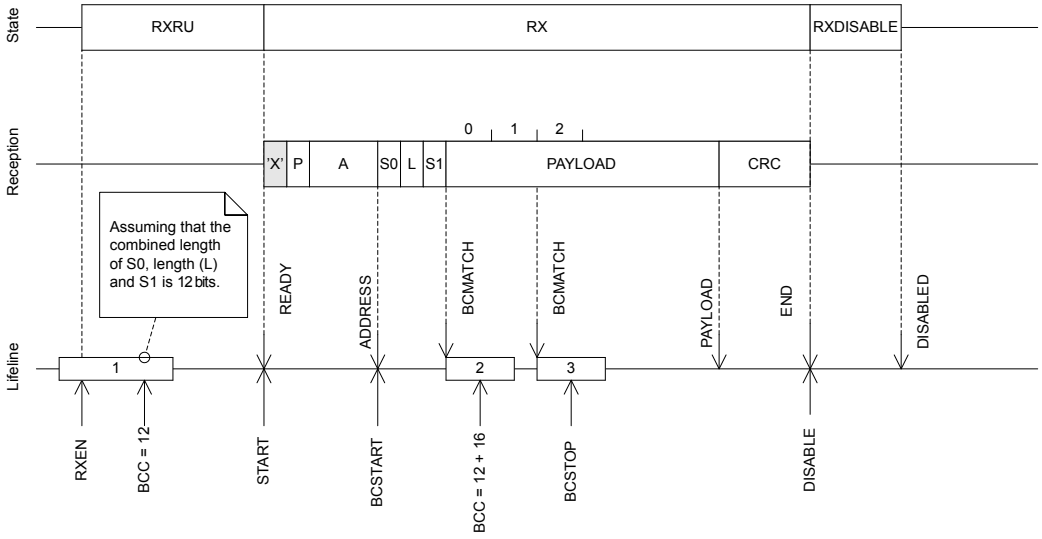



*Figure 70: Bit counter example*

## 6.14.12 Direction finding

The RADIO implements the Angle-of-Arrival (AoA) and Angle-of-Departure (AoD) Bluetooth Low Energy feature, which can be used to determine the direction of a peer device. The feature is available for the BLE 1 Mbps and BLE 2 Mbps modes.

When using this feature, the transmitter sends a packet with a continuous tone extension (CTE) appended to the packet, after the CRC. During the CTE, the receiver can take IQ samples of the incoming signal.

An antenna array is employed at the transmitter (AoD) or at the receiver (AoA). The AoD transmitter, or AoA receiver, switches between the antennas, in order to collect IQ samples from the different antenna pairs. The IQ samples can be used to calculate the relative path lengths between the antenna pairs, which can be used to estimate the direction of the transmitter.

### 6.14.12.1 CTE format

The CTE is from 16 µs to 160 µs and consists of an unwhitened sequence of 1's, equivalent to a continuous tone nominally offset from the carrier by +250 kHz for the 1 Mbps PHY and +500 kHz for the 2 Mbps BLE PHYs. The format of the CTE, when switching and/or sampling, is shown below.

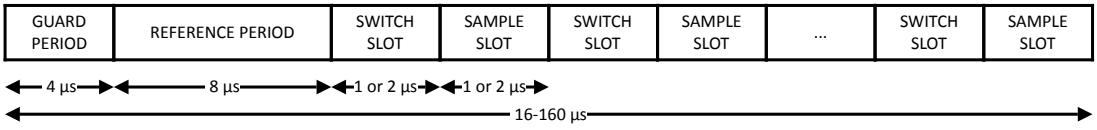



*Figure 71: Constant tone extension (CTE) structure*

Antenna switching is performed during switch slots and the guard period. The AoA/AoD feature requires that one IQ sample is taken for each microsecond within the reference period, and once for each sample

slot. Oversampling is possible by changing the sample spacing as described in IQ sampling on page 193. The switch slot and sample slot durations are either 1 or 2 µs, but must be equal. The format of the CTE and switching and sampling procedures may be configured prior to, or during, packet transmission and reception. Alternatively, during packet reception, these operations can be configured by reading specific fields of the packet contents.

### 6.14.12.2 Mode

Depending on the DFEMODE, the device performs the following procedures:

| | | DFEMODE | | | |
|---|---|---|---|---|---|
| | | AOA | | AOD | |
| | | TX | RX | TX | RX |
| AoA/AoD Procedure | Generating and transmitting CTE | x | | x | |
| | Receiving, interpreting, and sampling CTE | | x | | x |
| | Antenna switching | | x | x | |

*Table 61: AoA/AoD Procedures performed as a function of DFEMODE and TX/RX mode*

### 6.14.12.3 Inline configuration

When inline configuration is enabled during RX, further configuration of the AoA/AoD procedures is performed based on the values of the CP bit and the CTEInfo octet within the packet. This is enabled by setting CTEINLINECONF.CTEINLINECTRLEN. The CTEInfo octet is present only if the CP bit is set. The position of the CP bit and CTEInfo octet depends on whether the packet has a *Data Channel PDU* (CTEINLINECONF.CTEINFOINS1=InS1), or an *Advertising Channel PDU* (CTEINLINECONF.CTEINFOINS1=NotInS1).

#### Data channel PDU

For Data Channel PDUs, PCNF0.S0LEN must be 1 byte, and PCNF0.LFLEN must be 8 bits. To determine if S1 is present, the registers CTEINLINECONF.S0MASK and CTEINLINECONF.S0CONF forms a bitwise mask-and-test for the S0 field. If the bitwise AND between S0 and S0MASK equals S0CONF, then S1 is determined to be present. When present, the value of PCNF0.S1LEN will be ignored, as this is decided by the CP bit in the the following figure.

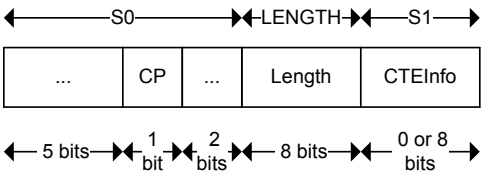



*Figure 72: Data channel PDU header*

When encrypting and decrypting BLE packets using the CCM peripheral, it is also required to set PCNF0.S1INCL=1. The CCM mode must be configured to use an 8-bit length field. The value of the CP bit is included in the calculation of the MIC, while the S1 field is ignored by the CCM calculation.

#### Advertising channel PDU

For advertising channel PDUs, the CTEInfo Flag replaces the CP bit. The CTEInfo Flag is within the extended header flag field in some of the advertising PDUs that employ the common extended advertising payload format (i.e. AUX_SYNC_IND, AUX_CHAIN_IND). The format of such packets is shown in the following figure.

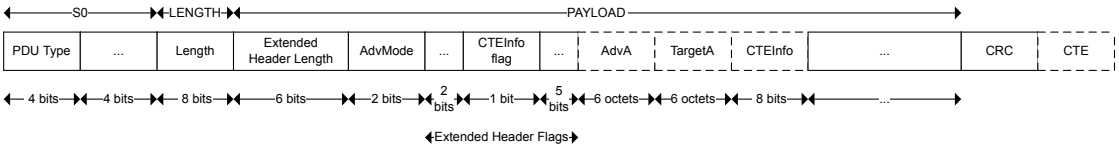



*Figure 73: Advertising channel PDU header*

The CTEINLINECONF.S0CONF and CTEINLINECONF.S0MASK fields can be configured to accept only certain advertising PDU Types. If the extended header length is non-zero, the CTEInfo extended header flag is checked to determine whether CTEInfo is present. If a bit before the CTEInfo flag within the extended header flags is set, then the CTEInfo position is postponed 6 octets.

### CTEInfo parsing

The CTEInfo field is shown in the following figure.

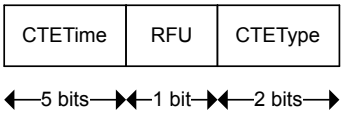



*Figure 74: CTEInfo field*

The CTETIME field defines the length of the CTE in 8 µs units. The valid upper bound of values can be adjusted using CTEINLINECONF.CTETIMEVALIDRANGE, including allowing use of the RFU bit within this field. If the CTETIME field is an invalid value of either 0 or 1, the CTE is assumed to be the minimum valid length of 16 µs. The slot duration is determined by the CTEType field. In RX this determines whether the sample spacing as defined in CTEINLINECONF.CTEINLINERXMODE1US or CTEINLINECONF.CTEINLINERXMODE2US is used.

| CTEType | Description | TX switch spacing | RX sample spacing during reference period | Sample spacing RX during reference period |
|---|---|---|---|---|
| 0 | AoA, no switching | - | TSAMPLESPACING1 | TSAMPLESPACING2 |
| 1 | AoD, 1 µs slots | 2 µs | TSAMPLESPACING1 | CTEINLINERXMODE1US |
| 2 | AoD, 2 µs slots | 4 µs | TSAMPLESPACING1 | CTEINLINERXMODE2US |
| 3 | Reserved for future use | | | |

*Table 62: Switching and sampling spacing based on CTEType*

## 6.14.12.4 Manual configuration

If CTEINLINECONF.CTEINLINECTRLEN is not set, then the packet is not parsed to determine the CTE parameters, and the antenna switching and sampling is controlled by other registers, see Antenna switching on page 192. The length of the CTE is given in 8 µs units by DFECTRL1.NUMBEROF8US. The start of the antenna switching and/or sampling (denoted as an AoA/AoD procedure), can be configured to start at some trigger with an additional offset. Using DFECTRL1.DFEINEXTENSION, the trigger can be configured to be the end of the CRC, or alternatively, the ADDRESS event. The additional offset for antenna switching is configured using DFECTRL2.TSWITCHOFFSET. Similarly, the additional offset for antenna sampling is configured using DFECTRL2.TSAMPLEOFFSET.

## 6.14.12.5 Receive- and transmit sequences

The addition of the CTE to the transmitted packet is illustrated in the following figure.

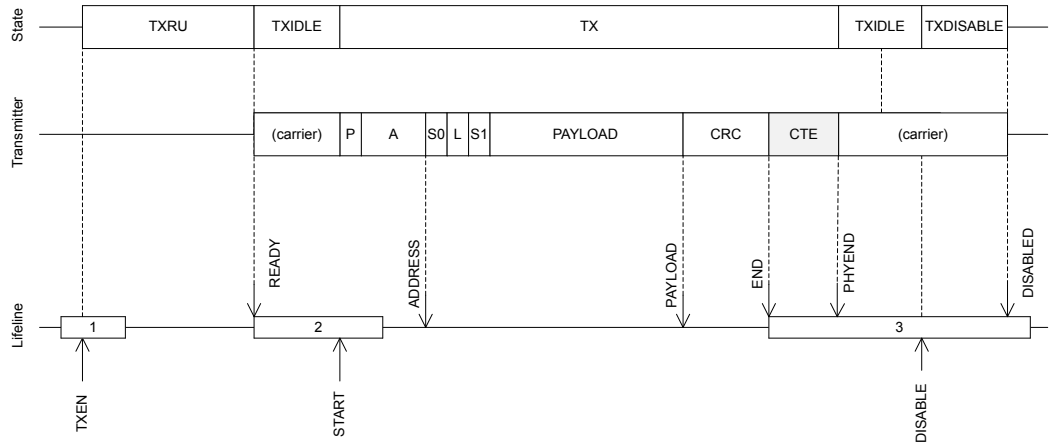



*Figure 75: Transmit sequence with DFE*

The prescence of CTE within a received packet is signalled by the CTEPRESENT and CTEWARNING events illustrated in the figure below.

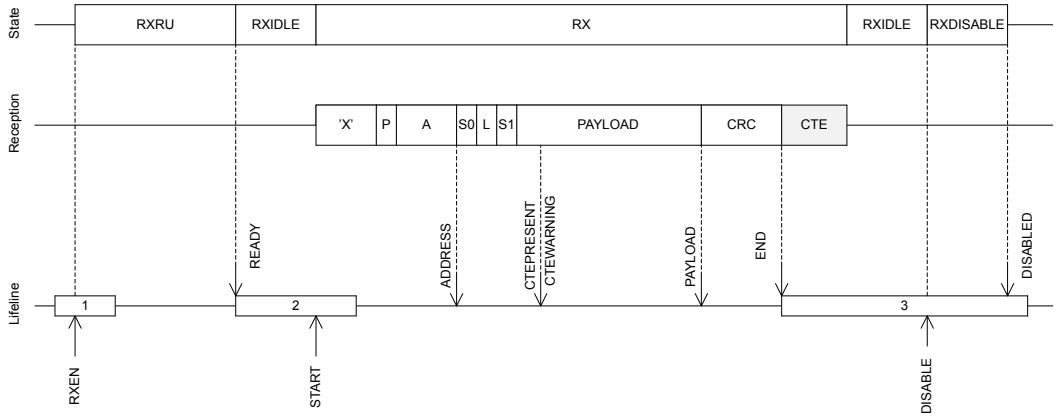



*Figure 76: Receive sequence with DFE*

## 6.14.12.6 Antenna switching

The RADIO can control up to 8 GPIO pins in order to control external antenna switches used in direction finding.

### Pin configuration

The eight antenna selection signals are mapped to physical pins according to the pin numbers specified in the PSEL.DFEGPIO[n] registers. Only pins that have the PSEL.DFEGPIO[n].CONNECTED field set to *Connected* will be controlled by the RADIO. Pins that are *Disconnected* will be controlled by GPIO.

During transmission in AoD TX mode or reception in AoA RX mode, the RADIO automatically acquires the pins as needed. At times when the RADIO does not use the pin, the pin is released to its default state and controlled by the GPIO configuration. Thus, the pin must be configured using the GPIO peripheral.

| Pin acquired by RADIO | Direction | Value | Comment |
|---|---|---|---|
| Yes | Output | Specified in SWITCHPATTERN | Pin acquired by RADIO, and in use for DFE. |
| No | Specified by GPIO | Specified by GPIO | DFE not in progress. Pin has not been acquired by RADIO, but is available for DFE use. |

*Table 63: Pin configuration matrix for a connected and enabled pin [n]*

### Switch pattern configuration

The values of the GPIOs while switching during the CTE are configured by writing successively to the SWITCHPATTERN register. The first write to SWITCHPATTERN is the GPIO pattern applied from the call of TASKS_TXEN or TASKS_RXEN until the first antenna switch is triggered. The second write sets the pattern for the reference period and is applied at the start of the guard period. The following writes set the pattern for the remaining switch slots and are applied at the start of each switch slot. If writing beyond the total number of antenna slots, the pattern will wrap to SWITCHPATTERN[2] and start over again. During operation, when the end of the SWITCHPATTERN buffer is reached, the RADIO cycles back to SWITCHPATTERN[2]. At the end of the AoA/AoD procedure, SWITCHPATTERN[0] is applied to DFECTRL1.TSWITCHSPACING after the previous antenna switch. The SWITCHPATTERN buffer can be erased/cleared using CLEARPATTERN.

A minimum number of three patterns must be written to the SWITCHPATTERN register.

If CTEINLINECONF.CTEINLINECTRLEN is not set, then the antenna switch spacing is determined by DFECTRL1.TSWITCHSPACING (otherwise described by Switching and sampling spacing based on CTEType on page 191). DFECTRL2.TSWITCHOFFSET determines the position of the first switch compared to the configurable start of CTE (see DFECTRL1.DFEINEXTENSION).

## 6.14.12.7 IQ sampling

The RADIO uses DMA to write IQ samples recorded during the CTE to RAM. Alternatively, the magnitude and phase of the samples can be recorded using the DFECTRL1.SAMPLETYPE field. The samples are written to the location in RAM specified by DFEPACKET.PTR. The maximum number of samples to transfer are specified by DFEPACKET.MAXCNT and the number of samples transferred are given in DFEPACKET.AMOUNT. The IQ samples are recorded with respect to the RX carrier frequency. The format of the samples is provided in the following table.

| SAMPLETYPE | Field | Bits | Description |
|---|---|---|---|
| 0: I_Q (default) | Q | 31:16 | 12 bits signed, sign extended to 16 bits |
| | I | 15:0 | |
| 1: MagPhase | reserved | 31:29 | Always zero |
| | magnitude | 28:16 | 13 bits unsigned. Equals 1.646756*sqrt(I^2+Q^2) |
| | phase | 15:0 | 9 bits signed, sign extended to 16 bits. Equals 64*atan2(Q, I) in the range [-201,201] |

*Table 64: Format of samples*

Oversampling is configured separately for the reference period and for the time after the reference period. During the reference period, the sample spacing is determined by DFECTRL1.TSAMPLESPACINGREF. DFECTRL2.TSAMPLEOFFSET determines the position of the first sample relative to the end of the last bit of the CRC.

For the time after the reference period, if CTEINLINECONF.CTEINLINECTRLEN is disabled, the sample spacing is set in DFECTRL1.TSAMPLESPACING. However, when CTEINLINECONF.CTEINLINECTRLEN is enabled, the sample spacing are determined by two different registers, depending on whether the device is in AoA or AoD RX-mode, as follows.

For AoD RX mode, the sample spacing after the reference period is determined by the CTEType in the packet, as listed in the table below.

| CTEType | Sample spacing |
|---|---|
| AoD 1 µs slots | CTEINLINECONF.CTEINLINERXMODE1US |
| AoD 2 µs slots | CTEINLINECONF.CTEINLINERXMODE2US |
| Other | DFECTRL1.TSAMPLESPACING |

*Table 65: Sample spacing when CTEINLINECONF.CTEINLINECTRLEN is set and the device is in AoD RX mode*

For AoA RX mode, the sample spacing after the reference period is determined by DFECTRL1.TSWITCHSPACING, as listed in the table below.

| DFECTRL1.TSWITCHSPACING | Sample spacing |
|---|---|
| 2 µs | CTEINLINECONF.CTEINLINERXMODE1US |
| 4 µs | CTEINLINECONF.CTEINLINERXMODE2US |
| Other | DFECTRL1.TSAMPLESPACING |

*Table 66: Sample spacing when CTEINLINECONF.CTEINLINECTRLEN is set and the device is in AoA RX mode*

For the reference- and switching periods, DFECTRL1.TSAMPLESPACINGREF and DFECTRL1.TSAMPLESPACING can be used to achieve oversampling.

## 6.14.13 IEEE 802.15.4 operation

With the MODE=Ieee802154_250kbit the RADIO will comply with the IEEE 802.15.4-2006 standard implementing its 250 kbps, 2450 MHz, O-QPSK PHY.

The IEEE 802.15.4 standard differs from Nordic's proprietary and *Bluetooth*® low energy modes. Notable differences include modulation scheme, channel structure, packet structure, security, and medium access control.

The main features of the IEEE 802.15.4 mode are:

- Ultra-low power 250 kbps, 2450 MHz, IEEE 802.15.4-2006 compliant link
- Clear channel assessment
- Energy detection scan
- CRC generation

### 6.14.13.1 Packet structure

The IEEE 802.15.4 standard defines an on-the-air frame/packet that is different from what is used in BLE mode.

The following figure provides an overview of the physical frame structure and its timing.

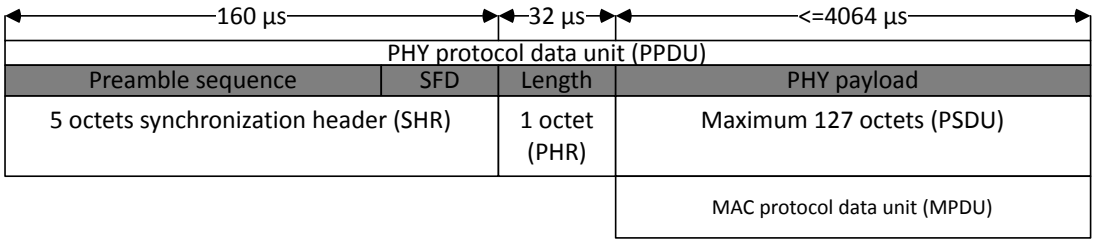



*Figure 77: IEEE 802.15.4 frame format (PPDU)*

The standard uses the term *octet* for an 8-bit storage unit within the PPDU. For timing, the value *symbol* is used, and it has a duration of 16 µs.

The total usable payload (PSDU) is 127 octets, but when CRC is in use, this is reduced to 125 octets of usable payload.

The preamble sequence consists of four octets that are all zero, and are used for synchronizing the RADIO's receiver. Following the preamble is the single octet *start of frame delimiter (SFD)*, with a fixed value of 0xA7. An alternate SFD can be programmed through the SFD register, providing an initial level of frame filtering for those who choose non-standard compliance. It is a valuable feature when operating in a congested or private network. The preamble sequence and the SFD are generated by the RADIO, and are not programmed by the user into the frame buffer.

Following the five octet *synchronization header (SHR)* is the single octet *phy header (PHR)*. The least significant seven bits of PHR denote the frame length of the following PSDU. The most significant bit is reserved and is set to zero for frames that are standard compliant. The RADIO reports all eight bits which can be used to carry additional information. The PHR is the first byte written to the frame data memory pointed to by PACKETPTR. Frames with zero length are discarded, and the FRAMESTART event is not generated in this case.

The next N octets carry the data of the PHY packet, where N equals the value of the PHR. For an implementation also using the IEEE 802.15.4 MAC layer, the PHY data is a MAC frame of N-2 octets, since two octets occupy a CRC field.

As illustrated in the figure below, an IEEE 802.15.4 MAC layer frame always consists of

- A header:
  - The frame control field (FCF)
  - The sequence number
  - Addressing fields
- A payload
- The 16-bit frame control sequence (FCS)

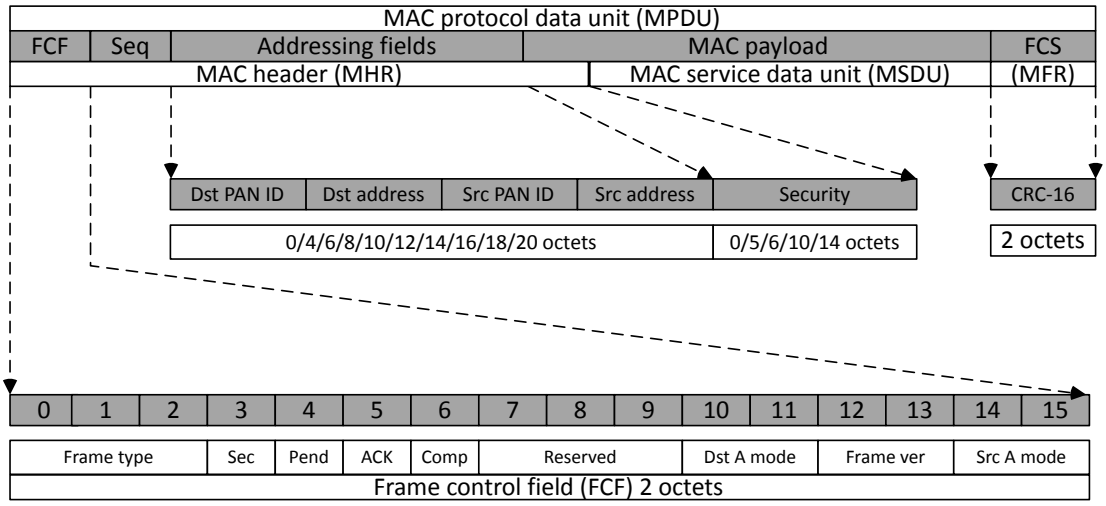



*Figure 78: IEEE 802.15.4 frame format (MPDU)*

The two FCF octets contain information about the frame type, addressing, and other control flags. This field is decoded when using the assisted operating modes offered by the RADIO.

The sequence number is a single octet in size and is unique for a frame. It is used in the associated acknowledgement frame sent upon successful frame reception.

The addressing field can be zero (acknowledgement frame) or up to 20 octets in size. The field is used to direct packets to the correct recipient and denote its origin. IEEE 802.15.4 bases its addressing on networks being organized in PANs with 16-bit identifier and nodes having a 16-bit or 64-bit address. In the assisted receive mode, these parameters are analyzed for address matching and acknowledgement.

The MAC payload carries the data of the next higher layer, or in the case of a MAC command frame, information used by the MAC layer itself.

The two last octets contain the 16-bit ITU-T CRC. The FCS is calculated over the MAC header (MHR) and MAC payload (MSDU) parts of the frame. This field is calculated automatically when sending a frame, or indicated in the CRCSTATUS register when a frame is received. If configured, this feature is taken care of autonomously by the CRC module.

### 6.14.13.2 Operating frequencies

The IEEE 802.15.4 standard defines 16 channels, 11 - 26, of 5 MHz each, in the 2450 MHz frequency band.

To choose the correct channel center frequency, the FREQUENCY register must be programmed according to the table below.

| IEEE 802.15.4 channel | Center frequency (MHz) | FREQUENCY setting |
|---|---|---|
| Channel 11 | 2405 | 5 |
| Channel 12 | 2410 | 10 |
| Channel 13 | 2415 | 15 |
| Channel 14 | 2420 | 20 |
| Channel 15 | 2425 | 25 |
| Channel 16 | 2430 | 30 |
| Channel 17 | 2435 | 35 |
| Channel 18 | 2440 | 40 |
| Channel 19 | 2445 | 45 |
| Channel 20 | 2450 | 50 |
| Channel 21 | 2455 | 55 |
| Channel 22 | 2460 | 60 |
| Channel 23 | 2465 | 65 |
| Channel 24 | 2470 | 70 |
| Channel 25 | 2475 | 75 |
| Channel 26 | 2480 | 80 |

*Table 67: IEEE 802.15.4 center frequency definition*

### 6.14.13.3 Energy detection (ED)

As required by the IEEE 802.15.4 standard, it must be possible to sample the received signal power within the bandwidth of a channel, for the purpose of determining presence of activity.

To prevent the channel signal from being decoded, the shortcut between the READY event and the START task should be disabled before putting the RADIO in receive mode. The energy detection (ED) measurement time, where RSSI samples are averaged, is 8 symbol periods, corresponding to 128 µs. The standard further specifies the measurement to be a number between 0 and 255, where 0 shall indicate received power less than 10 dB above the selected receiver sensitivity. The power range of the ED values must be at least a 40 dB linear mapping with accuracy of ±6 dB. See section *6.9.7 Receiver ED* in the IEEE 802.15.4 standard for further details.

The following example shows how to perform a single energy detection measurement and convert to IEEE 802.15.4 scale.

**IEEE 802.15.4 ED measurement example**

```
#define ED_RSSISCALE 4 // From electrical specifications
uint8_t sample_ed(void)
{
    int val;
    NRF_RADIO->TASKS_EDSTART = 1; // Start
    while (NRF_RADIO->EVENTS_EDEND != 1) {
        // CPU can sleep here or do something else
        // Use of interrupts are encouraged
        }
    val = NRF_RADIO->EDSAMPLE; // Read level
    return (uint8_t)(val>63 ? 255 : val*ED_RSSISCALE); // Convert to IEEE
 802.15.4 scale
}
```

For scaling between hardware value and dBm, see equation Conversion between hardware value and dBm on page 198.

The mlme-scan.req primitive of the MAC layer uses the ED measurement to detect channels where there might be wireless activity. To assist this primitive, a tailored mode of operation is available where the ED measurement runs for a defined number of iterations keeping track of the maximum ED level. This is enganged by writing the EDCNT register to a value different from 0, where it will run the specified number of iterations and report the maximum energy measurement in the EDSAMPLE register. The scan is started with EDSTART task and its end indicated with the EDEND event. This significantly reduces the interrupt frequency and therefore power consumption. The following figure shows how the ED measurement will operate depending on the EDCNT register.

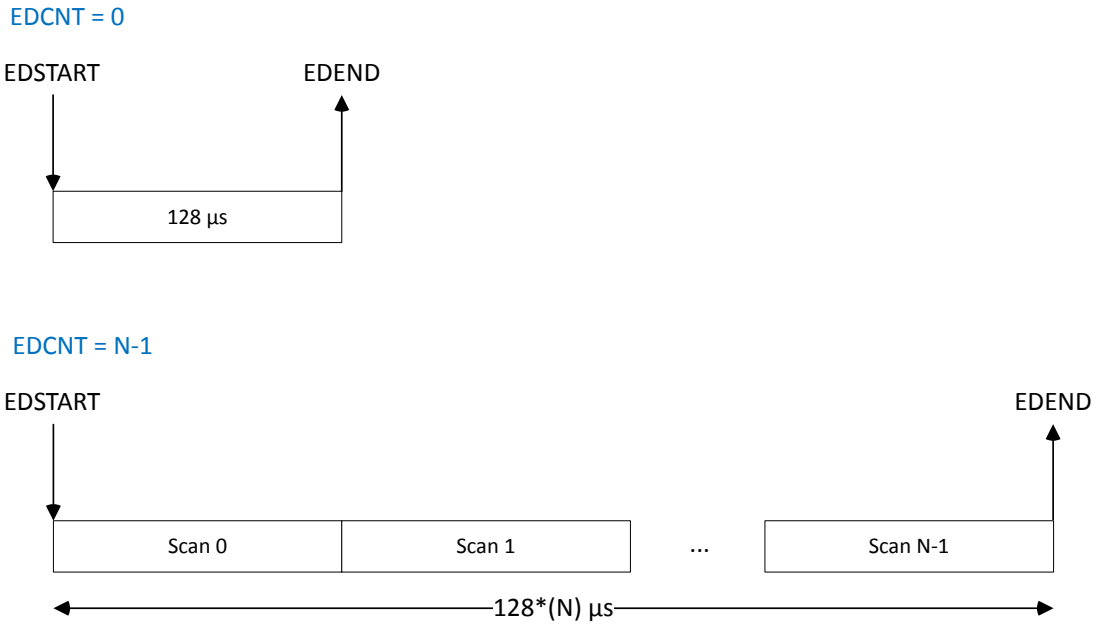



*Figure 79: Energy detection measurement examples*

The scan is stopped by writing the EDSTOP task. It will be followed by the EDSTOPPED event when the module has terminated.

## 6.14.13.4 Clear channel assessment (CCA)

IEEE 802.15.4 implements a listen-before-talk channel access method to avoid collisions when transmitting, known as *carrier sense multiple access with collision avoidance (CSMA-CA)*. The key part of this is measuring if the wireless medium is busy or not.

The following clear channel assesment modes are supported:

- *CCA Mode 1* (energy above threshold): The medium is reported busy upon detecting any energy above the ED threshold.
- *CCA Mode 2* (carrier sense only): The medium is reported busy upon detection of a signal compliant with the IEEE 802.15.4 standard with the same modulation and spreading characteristics.
- *CCA Mode 3* (carrier sense with energy above threshold): The medium is reported busy using a logical combination (AND/OR) between the results from CCA Mode 1 and CCA Mode 2.

The clear channel assessment should survey a period equal to 8 symbols or 128 µs.

The RADIO must be in receive mode and be able to receive correct packets when performing the CCA. The shortcut between READY and START must be disabled if baseband processing is not to be performed while the measurement is running.

### CCA Mode 1

*CCA Mode 1* is enabled by first configuring the field CCACTRL.CCAMODE=EdMode and writing the CCACTRL.CCAEDTHRES field to a chosen value. Once the CCASTART task is written, the RADIO will perform a ED measurement for 8 symbols and compare the measured level with that found in the CCACTRL.CCAEDTHRES field. If the measured value is higher than or equal to this threshold, the CCABUSY event is generated. If the measured level is less than the threshold, the CCAIDLE event is generated.

### CCA Mode 2

*CCA Mode 2* is enabled by configuring CCACTRL.CCAMODE=CarrierMode. The RADIO will sample to see if a valid SFD is found during the 8 symbols. If a valid SFD is detected, the CCABUSY event is generated and the device should not send any data. The CCABUSY event is also generated if the scan was performed during an ongoing frame reception. In the case where the measurement period completes with no SFD detection, the CCAIDLE event is generated. With CCACTRL.CCACORRCNT not being zero, the algorithm will look at the correlator output in addition to the SFD detection signal. If a SFD is reported during the scan period, it will terminate immidiately indicating busy medium. Similarly, if the number of peaks above CCACTRL.CCACORRTHRES crosses the CCACTRL.CCACORRCNT, the CCACTRL.CCABUSY event is generated. If less than CCACORRCOUNT crossings are found and no SFD is reported, the CCAIDLE event will be generated and the device can send data.

### CCA Mode 3

*CCA Mode 3* is enabled by configuring CCACTRL.CCAMODE=CarrierAndEdMode or CCACTRL.CCAMODE=CarrierOrEdMode, performing the required logical combination of the result from CCA Mode 1 and 2. The CCABUSY or CCAIDLE events are generated by ANDing or ORing the *energy above threshold* and *carrier detection* scans.

### Shortcuts

An ongoing CCA can always be stopped by issuing the CCASTOP task. This will trigger the associated CCASTOPPED event.

For CCA mode automation, a number of shortcuts are available.

- To automatically switch between RX (when performing the CCA) and to TX where the packet is sent, the shortcut between CCAIDLE and TXEN, in conjunction with the short between CCAIDLE and STOP muse be used.
- To automatically disable the RADIO whenever the CCA reports a busy medium, the shortcut between CCABUSY and DISABLE can be used.
- To immediately start a CCA after ramping up into RX mode, the shortcut between RXREADY and CCASTART can be used.

### Conversion

The conversion from a CCAEDTHRES, CCA, or EDLEVEL value to dBm can be done with the following equation, where $VAL_{HARDWARE}$ is the hardware-reported values, being either CCAEDTHRES, CCA or EDLEVEL, and constants ED_RSSISCALE and ED_RSSIOFFS are from electrical specifications.

$$P_{RF}[dBm] = ED\_RSSIOFFS + ED\_RSSISCALE \times VAL_{HARDWARE}$$

*Figure 80: Conversion between hardware value and dBm*

## 6.14.13.5 Cyclic redundancy check (CRC)

IEEE 802.15.4 uses a 16-bit ITU-T cyclic redundancy check (CRC) calculated over the MAC header (MHR) and MAC service data unit (MSDU).

The standard defines the following generator polynomial:

$G(x) = x^{16} + x^{12} + x^5 + 1$

In receive mode the RADIO will trigger the CRC module when the first octet after the frame length (PHR) is received. The CRC will then update on each consecutive octet received. When a complete frame is received the CRCSTATUS register will be updated accordingly and the CRCOK or CRCERROR events generated. When the CRC module is enabled it will not write the two last octets (CRC) to the frame Data RAM. When transmitting, the CRC will be computed on the fly, starting with the first octet after PHR, and inserted as the two last octets in the frame. The EasyDMA will fetch frame length minus 2 octets from RAM and insert the CRC octets insitu.

Below is a code snippet for configuring the CRC module for correct operation when in IEEE 802.15.4 mode. The CRCCNF is written to 16-bit CRC and the CRCPOLY is written to 0x11021. The start value used by IEEE 802.15.4 is zero and CRCINIT is configured to reflect this.

```
/* 16-bit CRC with ITU-T polynomial with 0 as start condition*/
NRF_RADIO->CRCCNF = ((RADIO_CRCCNF_SKIPADDR_Ieee802154 << RADIO_CRCCNF_SKIPADDR_Pos) |
                     (RADIO_CRCCNF_LEN_Two << RADIO_CRCCNF_LEN_Pos));
NRF_RADIO->CRCPOLY = 0x11021;
NRF_RADIO->CRCINIT = 0;
```

The ENDIANESS subregister must be set to little-endian since the FCS field is transmitted from left bit to right.

## 6.14.13.6 Transmit sequence

The transmission is started by first putting the RADIO in receive mode and triggering the RXEN task.

An outline of the IEEE 802.15.4 transmission is illustrated in the figure below.

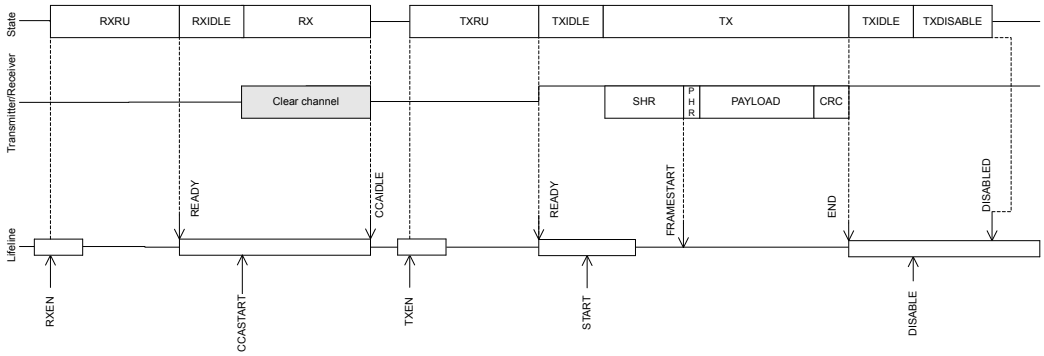



*Figure 81: IEEE 802.15.4 transmit sequence*

The receiver will ramp up and enter the RXIDLE state where the READY event is generated. Upon receiving the ready event, the CCA is started by triggering the CCASTART task. The chosen mode of assessment (CCACTRL.CCAMODE register) will be performed and signal the CCAIDLE or CCABUSY event 128 µs later. If the CCABUSY event is received, the RADIO will have to retry the CCA after a specific back-off period. This is outlined in the *IEEE 802.15.4 standard, Figure 69 in section 7.5.1.4 The CSMA-CA algorithm*.

If the CCAIDLE event is generated, a write to the TXEN task register enters the RADIO in TXRU state. The READY event will be generated when the RADIO is in TXIDLE state and ready to transmit. With the PACKETPTR pointing to the length (PHR) field of the frame, the START task can be written. The RADIO will send the four octet preamble sequence followed by the start of frame delimiter (SFD register). The first byte read from the Data RAM is the length field (PHR) followed by the transmission of the number of bytes indicated as the frame length. If the CRC module is configured it will run for PHR-2 octets. The last two octets will be substituted with the results from running the CRC. The necessary CRC parameters are sampled on the START task. The FCS field of the frame is little endian.

In addition to the already available shortcuts, one is provided between READY event and CCASTART task so that a CCA can automatically start when the receiver is ready. A second shortcut has been added between CCAIDLE event and the TXEN task, so that upon detecting a clear channel the RADIO can immediately enter transmit mode.

### 6.14.13.7 Receive sequence

The reception is started by first putting the RADIO in receive mode. After writing to the RXEN task, the RADIO will start ramping up and enter the RXRU state.

When the READY event is generated, the RADIO enters the RXIDLE mode. For the baseband processing to be enabled, the START task must be written. An outline of the IEEE 802.15.4 reception can be found in the figure below.

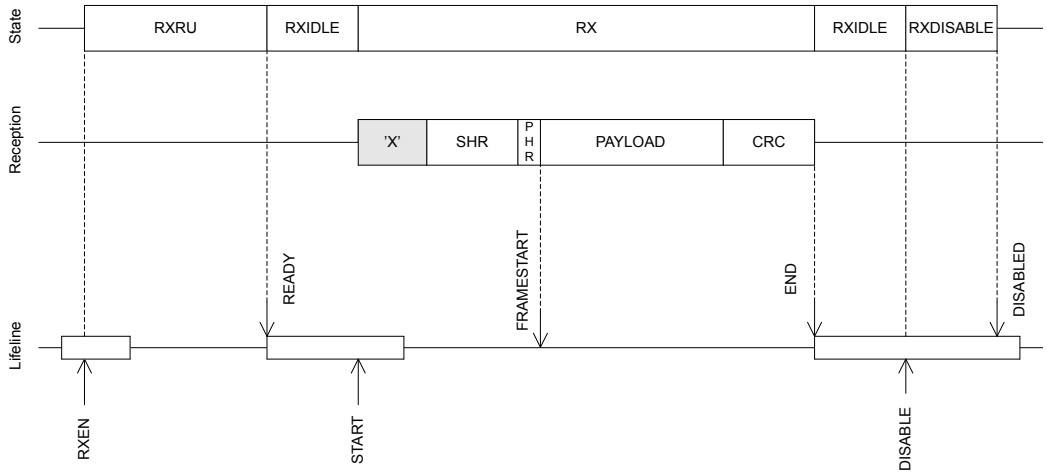



*Figure 82: IEEE 802.15.4 receive sequence*

When a valid SHR is received the RADIO will start storing future octets (starting with PHR) to the data memory pointed to by PACKETPTR. After the SFD octet is received the FRAMESTART event is generated. If the CRC module is enabled it will start updating with the second byte received (first byte in payload) and run for the full frame length. The two last bytes in the frame are not written to RAM when CRC is configured. However, if the result of the CRC after running the full frame is zero, the CRCOK event will be generated. The END event is generated when the last octet has been received and is available in data memory.

When a packet is received a link quality indicator (LQI) is also generated and appended immediately after the last received octet. When using an IEEE 802.15.4 compliant frame, this will be just after the MSDU since the FCS is not reported. In the case of a non-complient frame it will be appended after the full frame. The LQI reported by hardware must be converted to IEEE 802.15.4 range by an 8-bit saturating multiplication by 4, as shown in IEEE 802.15.4 ED measurement example on page 196. The LQI is only valid for frames equal to or longer than three octets. When receiving a frame the RSSI (reported as negative dB) will be measured at three points during the reception. These three values will be sorted and the middle one selected (median 3) to be remapped within the LQI range. The following figure illustrates the LQI measurement and how the data is arranged in data memory.

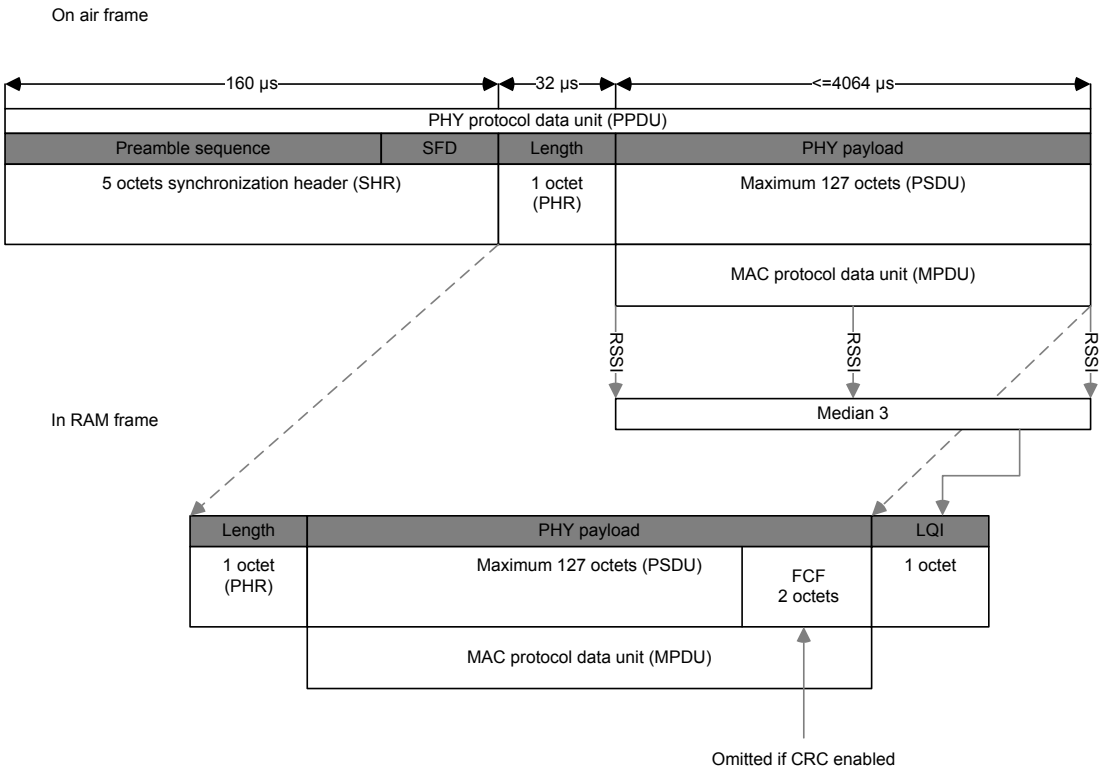



*Figure 83: IEEE 802.15.4 frame in data memory*

A shortcut has been added between the FRAMESTART event and the BCSTART task. This can be used to trigger a BCMATCH event after N bits, such as when inspecting the MAC addressing fields.

## 6.14.13.8 Interframe spacing (IFS)

The IEEE 802.15.4 standard defines a specific time that is alotted for the MAC sublayer to process received data. Interframe spacing (IFS) is used to prevent that two frames are transmitted too close together. If the transmission is requesting an acknowledgement, the space before the second frame shall be at least one IFS period.

The IFS is determined to be one of the following:

- IFS equals macMinSIFSPeriod (12 symbols) if the MPDU is less than or equal to aMaxSIFSFrameSize (18 octets) octets
- IFS equals macMinLIFSPeriod (40 symbols) if the MPDU is larger than aMaxSIFSFrameSize

Using the efficient assisted modes in the RADIO, the TIFS will be programmed with the correct value based on the frame being transmitted. If the assisted modes are not being used the user must update the TIFS register manually. The figure below provides details on what IFS period is valid in both acknowledged and unacknowledged transmissions.

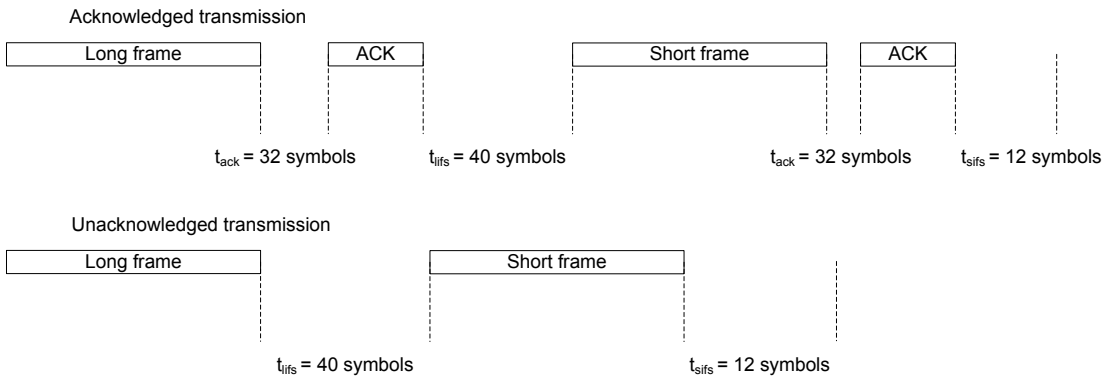



*Figure 84: Interframe spacing examples*

## 6.14.14 EasyDMA

The RADIO uses EasyDMA to read and write packets to RAM without CPU involvement.

As illustrated in RADIO block diagram on page 180, the RADIO's EasyDMA utilizes the same PACKETPTR for receiving and transmitting packets. This pointer should be reconfigured by the CPU each time before RADIO is started by the START task. The PACKETPTR register is double-buffered, meaning that it can be updated and prepared for the next transmission.

**Important:** If PACKETPTR is not pointing to the Data RAM region, an EasyDMA transfer may result in a HardFault or RAM corruption. See Memory on page 16 for more information about the different memory regions.

The END event indicates that the last bit has been processed by the RADIO. The DISABLED event is issued to acknowledge that a DISABLE task is done.

The structure of a packet is described in detail in Packet configuration on page 181. The data that is stored in Data RAM and transported by EasyDMA consists of the following fields:

- S0
- LENGTH
- S1
- PAYLOAD

In addition, a static add-on is sent immediately after the payload.

The size of each of the above fields in the frame is configurable (see Packet configuration on page 181), and the space occupied in RAM depends on these settings. The size of the field can be zero, as long as the resulting frame complies with the chosen RF protocol.

All fields are extended in size to align with a byte boundary in RAM. For instance, a 3-bit long field on air will occupy 1 byte in RAM while a 9-bit long field will be extended to 2 bytes.

The packet's elements can be configured as follows:

- CI, TERM1, and TERM2 fields are only present in *Bluetooth*® low energy long range mode
- S0 is configured through the PCNF0.S0LEN field
- LENGTH is configured through the PCNF0.LFLEN field
- S1 is configured through the PCNF0.S1LEN field
- Payload size is configured through the value in RAM corresponding to the LENGTH field
- Static add-on size is configured through the PCNF1.STATLEN field

The PCNF1.MAXLEN field configures the maximum packet payload plus add-on size in number of bytes that can be transmitted or received by the RADIO. This feature can be used to ensure that the RADIO does not overwrite, or read beyond, the RAM assigned to the packet payload. This means that if the LENGTH

field of the packet payload exceedes PCNF1.STATLEN, and the LENGTH field in the packet specifies a packet larger than configured in PCNF1.MAXLEN, the payload will be truncated to the length specified in PCNF1.MAXLEN.

**Note:** The PCNF1.MAXLEN field includes the payload and the add-on, but excludes the size occupied by the S0, LENGTH, and S1 fields. This has to be taken into account when allocating RAM.

If the payload and add-on length is specified larger than PCNF1.MAXLEN, the RADIO will still transmit or receive in the same way as before, except the payload is now truncated to PCNF1.MAXLEN. The packet's LENGTH field will not be altered when the payload is truncated. The RADIO will calculate CRC as if the packet length is equal to PCNF1.MAXLEN.

**Note:** If PACKETPTR is not pointing to the Data RAM region, an EasyDMA transfer may result in a HardFault or RAM corruption. See Memory on page 16 for more information about the different memory regions.

The END event indicates that the last bit has been processed by the RADIO. The DISABLED event is issued to acknowledge that an DISABLE task is done.

## 6.14.15 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40001000 | RADIO | RADIO | 2.4 GHz radio | |

*Table 68: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_TXEN | 0x000 | Enable RADIO in TX mode |
| TASKS_RXEN | 0x004 | Enable RADIO in RX mode |
| TASKS_START | 0x008 | Start RADIO |
| TASKS_STOP | 0x00C | Stop RADIO |
| TASKS_DISABLE | 0x010 | Disable RADIO |
| TASKS_RSSISTART | 0x014 | Start the RSSI and take one single sample of the receive signal strength |
| TASKS_RSSISTOP | 0x018 | Stop the RSSI measurement |
| TASKS_BCSTART | 0x01C | Start the bit counter |
| TASKS_BCSTOP | 0x020 | Stop the bit counter |
| TASKS_EDSTART | 0x024 | Start the energy detect measurement used in IEEE 802.15.4 mode |
| TASKS_EDSTOP | 0x028 | Stop the energy detect measurement |
| TASKS_CCASTART | 0x02C | Start the clear channel assessment used in IEEE 802.15.4 mode |
| TASKS_CCASTOP | 0x030 | Stop the clear channel assessment |
| EVENTS_READY | 0x100 | RADIO has ramped up and is ready to be started |
| EVENTS_ADDRESS | 0x104 | Address sent or received |
| EVENTS_PAYLOAD | 0x108 | Packet payload sent or received |
| EVENTS_END | 0x10C | Packet sent or received |
| EVENTS_DISABLED | 0x110 | RADIO has been disabled |
| EVENTS_DEVMATCH | 0x114 | A device address match occurred on the last received packet |
| EVENTS_DEVMISS | 0x118 | No device address match occurred on the last received packet |
| EVENTS_RSSIEND | 0x11C | Sampling of receive signal strength complete |
| EVENTS_BCMATCH | 0x128 | Bit counter reached bit count value |
| EVENTS_CRCOK | 0x130 | Packet received with CRC ok |
| EVENTS_CRCERROR | 0x134 | Packet received with CRC error |
| EVENTS_FRAMESTART | 0x138 | IEEE 802.15.4 length field received |

| Register | Offset | Description |
|---|---|---|
| EVENTS_EDEND | 0x13C | Sampling of energy detection complete. A new ED sample is ready for readout from the RADIO.EDSAMPLE register. |
| EVENTS_EDSTOPPED | 0x140 | The sampling of energy detection has stopped |
| EVENTS_CCAIDLE | 0x144 | Wireless medium in idle - clear to send |
| EVENTS_CCABUSY | 0x148 | Wireless medium busy - do not send |
| EVENTS_CCASTOPPED | 0x14C | The CCA has stopped |
| EVENTS_RATEBOOST | 0x150 | Ble_LR CI field received, receive mode is changed from Ble_LR125Kbit to Ble_LR500Kbit. |
| EVENTS_TXREADY | 0x154 | RADIO has ramped up and is ready to be started TX path |
| EVENTS_RXREADY | 0x158 | RADIO has ramped up and is ready to be started RX path |
| EVENTS_MHRMATCH | 0x15C | MAC header match found |
| EVENTS_PHYEND | 0x16C | Generated in Ble_LR125Kbit, Ble_LR500Kbit and Ieee802154_250Kbit modes when last bit is sent on air. |
| EVENTS_CTEPRESENT | 0x170 | CTE is present (early warning right after receiving CTEInfo byte) |
| SHORTS | 0x200 | Shortcuts between local events and tasks |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| CRCSTATUS | 0x400 | CRC status |
| RXMATCH | 0x408 | Received address |
| RXCRC | 0x40C | CRC field of previously received packet |
| DAI | 0x410 | Device address match index |
| PDUSTAT | 0x414 | Payload status |
| CTESTATUS | 0x44C | CTEInfo parsed from received packet |
| DFESTATUS | 0x458 | DFE status information |
| PACKETPTR | 0x504 | Packet pointer |
| FREQUENCY | 0x508 | Frequency |
| TXPOWER | 0x50C | Output power |
| MODE | 0x510 | Data rate and modulation |
| PCNF0 | 0x514 | Packet configuration register 0 |
| PCNF1 | 0x518 | Packet configuration register 1 |
| BASE0 | 0x51C | Base address 0 |
| BASE1 | 0x520 | Base address 1 |
| PREFIX0 | 0x524 | Prefixes bytes for logical addresses 0-3 |
| PREFIX1 | 0x528 | Prefixes bytes for logical addresses 4-7 |
| TXADDRESS | 0x52C | Transmit address select |
| RXADDRESSES | 0x530 | Receive address select |
| CRCCNF | 0x534 | CRC configuration |
| CRCPOLY | 0x538 | CRC polynomial |
| CRCINIT | 0x53C | CRC initial value |
| TIFS | 0x544 | Interframe spacing in µs |
| RSSISAMPLE | 0x548 | RSSI sample |
| STATE | 0x550 | Current radio state |
| DATAWHITEIV | 0x554 | Data whitening initial value |
| BCC | 0x560 | Bit counter compare |
| DAB[n] | 0x600 | Device address base segment n |
| DAP[n] | 0x620 | Device address prefix n |
| DACNF | 0x640 | Device address match configuration |
| MHRMATCHCONF | 0x644 | Search pattern configuration |
| MHRMATCHMAS | 0x648 | Pattern mask |
| MODECNF0 | 0x650 | Radio mode configuration register 0 |
| SFD | 0x660 | IEEE 802.15.4 start of frame delimiter |
| EDCNT | 0x664 | IEEE 802.15.4 energy detect loop count |
| EDSAMPLE | 0x668 | IEEE 802.15.4 energy detect level |
| CCACTRL | 0x66C | IEEE 802.15.4 clear channel assessment control |

| Register | Offset | Description |
|---|---|---|
| DFEMODE | 0x900 | Whether to use Angle-of-Arrival (AOA) or Angle-of-Departure (AOD) |
| CTEINLINECONF | 0x904 | Configuration for CTE inline mode |
| DFECTRL1 | 0x910 | Various configuration for Direction finding |
| DFECTRL2 | 0x914 | Start offset for Direction finding |
| SWITCHPATTERN | 0x928 | GPIO patterns to be used for each antenna |
| CLEARPATTERN | 0x92C | Clear the GPIO pattern array for antenna control |
| PSEL.DFEGPIO[0] | 0x930 | Pin select for DFE pin 0 |
| PSEL.DFEGPIO[1] | 0x934 | Pin select for DFE pin 1 |
| PSEL.DFEGPIO[2] | 0x938 | Pin select for DFE pin 2 |
| PSEL.DFEGPIO[3] | 0x93C | Pin select for DFE pin 3 |
| PSEL.DFEGPIO[4] | 0x940 | Pin select for DFE pin 4 |
| PSEL.DFEGPIO[5] | 0x944 | Pin select for DFE pin 5 |
| PSEL.DFEGPIO[6] | 0x948 | Pin select for DFE pin 6 |
| PSEL.DFEGPIO[7] | 0x94C | Pin select for DFE pin 7 |
| DFEPACKET.PTR | 0x950 | Data pointer |
| DFEPACKET.MAXCNT | 0x954 | Maximum number of buffer words to transfer |
| DFEPACKET.AMOUNT | 0x958 | Number of samples transferred in the last transaction |
| POWER | 0xFFC | Peripheral power control |

*Table 69: Register overview*

## 6.14.15.1 TASKS_TXEN

Address offset: 0x000

Enable RADIO in TX mode

| Bit number | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | A |
| **Reset 0x00000000** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_TXEN | | | Enable RADIO in TX mode |
| | | | Trigger | 1 | Trigger task |

## 6.14.15.2 TASKS_RXEN

Address offset: 0x004

Enable RADIO in RX mode

| Bit number | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | A |
| **Reset 0x00000000** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_RXEN | | | Enable RADIO in RX mode |
| | | | Trigger | 1 | Trigger task |

## 6.14.15.3 TASKS_START

Address offset: 0x008

Start RADIO

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_START | | | Start RADIO |
| | | | Trigger | 1 | Trigger task |

## 6.14.15.4 TASKS_STOP

Address offset: 0x00C

Stop RADIO

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STOP | | | Stop RADIO |
| | | | Trigger | 1 | Trigger task |

## 6.14.15.5 TASKS_DISABLE

Address offset: 0x010

Disable RADIO

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_DISABLE | | | Disable RADIO |
| | | | Trigger | 1 | Trigger task |

## 6.14.15.6 TASKS_RSSISTART

Address offset: 0x014

Start the RSSI and take one single sample of the receive signal strength

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_RSSISTART | | | Start the RSSI and take one single sample of the receive signal strength |
| | | | Trigger | 1 | Trigger task |

## 6.14.15.7 TASKS_RSSISTOP

Address offset: 0x018

Stop the RSSI measurement

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_RSSISTOP | | | Stop the RSSI measurement |
| | | | Trigger | 1 | Trigger task |

## 6.14.15.8 TASKS_BCSTART

Address offset: 0x01C

Start the bit counter

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_BCSTART | | | Start the bit counter |
| | | | Trigger | 1 | Trigger task |

## 6.14.15.9 TASKS_BCSTOP

Address offset: 0x020

Stop the bit counter

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_BCSTOP | | | Stop the bit counter |
| | | | Trigger | 1 | Trigger task |

## 6.14.15.10 TASKS_EDSTART

Address offset: 0x024

Start the energy detect measurement used in IEEE 802.15.4 mode

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_EDSTART | | | Start the energy detect measurement used in IEEE 802.15.4 mode |
| | | | Trigger | 1 | Trigger task |

## 6.14.15.11 TASKS_EDSTOP

Address offset: 0x028

Stop the energy detect measurement

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_EDSTOP | | | Stop the energy detect measurement |
| | | | Trigger | 1 | Trigger task |

## 6.14.15.12 TASKS_CCASTART

Address offset: 0x02C

Start the clear channel assessment used in IEEE 802.15.4 mode

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_CCASTART | | | Start the clear channel assessment used in IEEE 802.15.4 mode |
| | | | Trigger | 1 | Trigger task |

## 6.14.15.13 TASKS_CCASTOP

Address offset: 0x030

Stop the clear channel assessment

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_CCASTOP | | | Stop the clear channel assessment |
| | | | Trigger | 1 | Trigger task |

## 6.14.15.14 EVENTS_READY

Address offset: 0x100

RADIO has ramped up and is ready to be started

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_READY | | | RADIO has ramped up and is ready to be started |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.15 EVENTS_ADDRESS

Address offset: 0x104

Address sent or received

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_ADDRESS | | | Address sent or received |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.16 EVENTS_PAYLOAD

Address offset: 0x108

Packet payload sent or received

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_PAYLOAD | | | Packet payload sent or received |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.17 EVENTS_END

Address offset: 0x10C

Packet sent or received

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_END | | | Packet sent or received |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.18 EVENTS_DISABLED

Address offset: 0x110

RADIO has been disabled

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_DISABLED | | | RADIO has been disabled |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.19 EVENTS_DEVMATCH

Address offset: 0x114

A device address match occurred on the last received packet

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_DEVMATCH | | | A device address match occurred on the last received packet |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.20 EVENTS_DEVMISS

Address offset: 0x118

No device address match occurred on the last received packet

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_DEVMISS | | | No device address match occurred on the last received packet |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.21 EVENTS_RSSIEND

Address offset: 0x11C

Sampling of receive signal strength complete

A new RSSI sample is ready for readout from the RADIO.RSSISAMPLE register

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_RSSIEND | | | Sampling of receive signal strength complete<br>A new RSSI sample is ready for readout from the RADIO.RSSISAMPLE register |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.22 EVENTS_BCMATCH

Address offset: 0x128

Bit counter reached bit count value

Bit counter value is specified in the RADIO.BCC register

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_BCMATCH | | | Bit counter reached bit count value |
| | | | | | Bit counter value is specified in the RADIO.BCC register |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.23 EVENTS_CRCOK

Address offset: 0x130

Packet received with CRC ok

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_CRCOK | | | Packet received with CRC ok |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.24 EVENTS_CRCERROR

Address offset: 0x134

Packet received with CRC error

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_CRCERROR | | | Packet received with CRC error |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.25 EVENTS_FRAMESTART

Address offset: 0x138

IEEE 802.15.4 length field received

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_FRAMESTART | | | IEEE 802.15.4 length field received |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.26 EVENTS_EDEND

Address offset: 0x13C

Sampling of energy detection complete. A new ED sample is ready for readout from the RADIO.EDSAMPLE register.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_EDEND | | | Sampling of energy detection complete. A new ED sample is ready for readout from the RADIO.EDSAMPLE register. |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.27 EVENTS_EDSTOPPED

Address offset: 0x140

The sampling of energy detection has stopped

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_EDSTOPPED | | | The sampling of energy detection has stopped |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.28 EVENTS_CCAIDLE

Address offset: 0x144

Wireless medium in idle - clear to send

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_CCAIDLE | | | Wireless medium in idle - clear to send |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.29 EVENTS_CCABUSY

Address offset: 0x148

Wireless medium busy - do not send

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_CCABUSY | | | Wireless medium busy - do not send |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.30 EVENTS_CCASTOPPED

Address offset: 0x14C

The CCA has stopped

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_CCASTOPPED | | | The CCA has stopped |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.31 EVENTS_RATEBOOST

Address offset: 0x150

Ble_LR CI field received, receive mode is changed from Ble_LR125Kbit to Ble_LR500Kbit.

Ble_LR CI field received, receive mode is changed from Ble_LR125Kbit to Ble_LR500Kbit.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_RATEBOOST | | | Ble_LR CI field received, receive mode is changed from Ble_LR125Kbit to Ble_LR500Kbit.<br>Ble_LR CI field received, receive mode is changed from Ble_LR125Kbit to Ble_LR500Kbit. |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.32 EVENTS_TXREADY

Address offset: 0x154

RADIO has ramped up and is ready to be started TX path

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_TXREADY | | | RADIO has ramped up and is ready to be started TX path |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.33 EVENTS_RXREADY

Address offset: 0x158

RADIO has ramped up and is ready to be started RX path

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_RXREADY | | | RADIO has ramped up and is ready to be started RX path |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.34 EVENTS_MHRMATCH

Address offset: 0x15C

MAC header match found

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_MHRMATCH | | | MAC header match found |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.35 EVENTS_PHYEND

Address offset: 0x16C

Generated in Ble_LR125Kbit, Ble_LR500Kbit and Ieee802154_250Kbit modes when last bit is sent on air.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_PHYEND | | | Generated in Ble_LR125Kbit, Ble_LR500Kbit and Ieee802154_250Kbit modes when last bit is sent on air. |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.36 EVENTS_CTEPRESENT

Address offset: 0x170

CTE is present (early warning right after receiving CTEInfo byte)

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_CTEPRESENT | | | CTE is present (early warning right after receiving CTEInfo byte) |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.14.15.37 SHORTS

Address offset: 0x200

## Shortcuts between local events and tasks

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | U T S R Q P O N M L K H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | READY_START | | | Shortcut between event READY and task START |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| B | RW | END_DISABLE | | | Shortcut between event END and task DISABLE |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| C | RW | DISABLED_TXEN | | | Shortcut between event DISABLED and task TXEN |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| D | RW | DISABLED_RXEN | | | Shortcut between event DISABLED and task RXEN |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| E | RW | ADDRESS_RSSISTART | | | Shortcut between event ADDRESS and task RSSISTART |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| F | RW | END_START | | | Shortcut between event END and task START |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| G | RW | ADDRESS_BCSTART | | | Shortcut between event ADDRESS and task BCSTART |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| H | RW | DISABLED_RSSISTOP | | | Shortcut between event DISABLED and task RSSISTOP |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| K | RW | RXREADY_CCASTART | | | Shortcut between event RXREADY and task CCASTART |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| L | RW | CCAIDLE_TXEN | | | Shortcut between event CCAIDLE and task TXEN |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| M | RW | CCABUSY_DISABLE | | | Shortcut between event CCABUSY and task DISABLE |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| N | RW | FRAMESTART_BCSTART | | | Shortcut between event FRAMESTART and task BCSTART |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| O | RW | READY_EDSTART | | | Shortcut between event READY and task EDSTART |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| P | RW | EDEND_DISABLE | | | Shortcut between event EDEND and task DISABLE |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| Q | RW | CCAIDLE_STOP | | | Shortcut between event CCAIDLE and task STOP |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| R | RW | TXREADY_START | | | Shortcut between event TXREADY and task START |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | U T S R Q P O N M L K H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| S | RW | RXREADY_START | | | Shortcut between event RXREADY and task START |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| T | RW | PHYEND_DISABLE | | | Shortcut between event PHYEND and task DISABLE |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| U | RW | PHYEND_START | | | Shortcut between event PHYEND and task START |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |

## 6.14.15.38 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | a Z V U T S R Q P O N M L K I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | READY | | | Write '1' to enable interrupt for event READY |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | ADDRESS | | | Write '1' to enable interrupt for event ADDRESS |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | PAYLOAD | | | Write '1' to enable interrupt for event PAYLOAD |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | END | | | Write '1' to enable interrupt for event END |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E | RW | DISABLED | | | Write '1' to enable interrupt for event DISABLED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| F | RW | DEVMATCH | | | Write '1' to enable interrupt for event DEVMATCH |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| G | RW | DEVMISS | | | Write '1' to enable interrupt for event DEVMISS |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | a Z V U T S R Q P O N M L K I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| H | RW | RSSIEND | | | Write '1' to enable interrupt for event RSSIEND<br>A new RSSI sample is ready for readout from the RADIO.RSSISAMPLE register |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| I | RW | BCMATCH | | | Write '1' to enable interrupt for event BCMATCH<br>Bit counter value is specified in the RADIO.BCC register |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| K | RW | CRCOK | | | Write '1' to enable interrupt for event CRCOK |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| L | RW | CRCERROR | | | Write '1' to enable interrupt for event CRCERROR |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| M | RW | FRAMESTART | | | Write '1' to enable interrupt for event FRAMESTART |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| N | RW | EDEND | | | Write '1' to enable interrupt for event EDEND |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| O | RW | EDSTOPPED | | | Write '1' to enable interrupt for event EDSTOPPED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| P | RW | CCAIDLE | | | Write '1' to enable interrupt for event CCAIDLE |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| Q | RW | CCABUSY | | | Write '1' to enable interrupt for event CCABUSY |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| R | RW | CCASTOPPED | | | Write '1' to enable interrupt for event CCASTOPPED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| S | RW | RATEBOOST | | | Write '1' to enable interrupt for event RATEBOOST<br>Ble_LR CI field received, receive mode is changed from Ble_LR125Kbit to Ble_LR500Kbit. |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | a Z V U T S R Q P O N M L K I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| T | RW | TXREADY | | | Write '1' to enable interrupt for event TXREADY |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| U | RW | RXREADY | | | Write '1' to enable interrupt for event RXREADY |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| V | RW | MHRMATCH | | | Write '1' to enable interrupt for event MHRMATCH |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| Z | RW | PHYEND | | | Write '1' to enable interrupt for event PHYEND |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| a | RW | CTEPRESENT | | | Write '1' to enable interrupt for event CTEPRESENT |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.14.15.39 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | a Z V U T S R Q P O N M L K I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | READY | | | Write '1' to disable interrupt for event READY |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | ADDRESS | | | Write '1' to disable interrupt for event ADDRESS |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | PAYLOAD | | | Write '1' to disable interrupt for event PAYLOAD |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | END | | | Write '1' to disable interrupt for event END |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E | RW | DISABLED | | | Write '1' to disable interrupt for event DISABLED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | a Z V U T S R Q P O N M L K I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| | | | Enabled | 1 | Read: Enabled |
| F | RW | DEVMATCH | | | Write '1' to disable interrupt for event DEVMATCH |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| G | RW | DEVMISS | | | Write '1' to disable interrupt for event DEVMISS |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| H | RW | RSSIEND | | | Write '1' to disable interrupt for event RSSIEND<br>A new RSSI sample is ready for readout from the RADIO.RSSISAMPLE register |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| I | RW | BCMATCH | | | Write '1' to disable interrupt for event BCMATCH<br>Bit counter value is specified in the RADIO.BCC register |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| K | RW | CRCOK | | | Write '1' to disable interrupt for event CRCOK |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| L | RW | CRCERROR | | | Write '1' to disable interrupt for event CRCERROR |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| M | RW | FRAMESTART | | | Write '1' to disable interrupt for event FRAMESTART |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| N | RW | EDEND | | | Write '1' to disable interrupt for event EDEND |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| O | RW | EDSTOPPED | | | Write '1' to disable interrupt for event EDSTOPPED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| P | RW | CCAIDLE | | | Write '1' to disable interrupt for event CCAIDLE |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| Q | RW | CCABUSY | | | Write '1' to disable interrupt for event CCABUSY |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| R | RW | CCASTOPPED | | | Write '1' to disable interrupt for event CCASTOPPED |
| | | | Clear | 1 | Disable |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | a Z V U T S R Q P O N M L K I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| S | RW | RATEBOOST | | | Write '1' to disable interrupt for event RATEBOOST<br><br>Ble_LR CI field received, receive mode is changed from Ble_LR125Kbit to Ble_LR500Kbit. |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| T | RW | TXREADY | | | Write '1' to disable interrupt for event TXREADY |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| U | RW | RXREADY | | | Write '1' to disable interrupt for event RXREADY |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| V | RW | MHRMATCH | | | Write '1' to disable interrupt for event MHRMATCH |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| Z | RW | PHYEND | | | Write '1' to disable interrupt for event PHYEND |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| a | RW | CTEPRESENT | | | Write '1' to disable interrupt for event CTEPRESENT |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.14.15.40 CRCSTATUS

Address offset: 0x400

CRC status

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | CRCSTATUS | | | CRC status of packet received |
| | | | CRCError | 0 | Packet received with CRC error |
| | | | CRCOk | 1 | Packet received with CRC ok |

## 6.14.15.41 RXMATCH

Address offset: 0x408

Received address

| Bit number | 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 |
|---|---|
| ID | A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | RXMATCH | | | Received address |
| | | | | | Logical address of which previous packet was received |

## 6.14.15.42 RXCRC

Address offset: 0x40C

CRC field of previously received packet

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | RXCRC | | | CRC field of previously received packet |
| | | | | | CRC field of previously received packet |

## 6.14.15.43 DAI

Address offset: 0x410

Device address match index

| Bit number | 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 |
|---|---|
| ID | A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | DAI | | | Device address match index |
| | | | | | Index (n) of device address, see DAB[n] and DAP[n], that got an address match |

## 6.14.15.44 PDUSTAT

Address offset: 0x414

Payload status

| Bit number | 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 |
|---|---|
| ID | B B A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | PDUSTAT | | | Status on payload length vs. PCNF1.MAXLEN |
| | | | LessThan | 0 | Payload less than PCNF1.MAXLEN |
| | | | GreaterThan | 1 | Payload greater than PCNF1.MAXLEN |
| B | R | CISTAT | | | Status on what rate packet is received with in Long Range |
| | | | LR125kbit | 0 | Frame is received at 125 kbps |
| | | | LR500kbit | 1 | Frame is received at 500 kbps |

## 6.14.15.45 CTESTATUS

Address offset: 0x44C

CTEInfo parsed from received packet

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | C C B A A A A A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | CTETIME | | | CTETime parsed from packet |
| B | R | RFU | | | RFU parsed from packet |
| C | R | CTETYPE | | | CTEType parsed from packet |

## 6.14.15.46 DFESTATUS

Address offset: 0x458

DFE status information

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | B A A A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | SWITCHINGSTATE | | | Internal state of switching state machine |
| | | | Idle | 0 | Switching state Idle |
| | | | Offset | 1 | Switching state Offset |
| | | | Guard | 2 | Switching state Guard |
| | | | Ref | 3 | Switching state Ref |
| | | | Switching | 4 | Switching state Switching |
| | | | Ending | 5 | Switching state Ending |
| B | R | SAMPLINGSTATE | | | Internal state of sampling state machine |
| | | | Idle | 0 | Sampling state Idle |
| | | | Sampling | 1 | Sampling state Sampling |

## 6.14.15.47 PACKETPTR

Address offset: 0x504

Packet pointer

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PACKETPTR | | | Packet pointer |
| | | | | | Packet address to be used for the next transmission or reception. When transmitting, the packet pointed to by this address will be transmitted and when receiving, the received packet will be written to this address. This address is a byte aligned RAM address. |
| | | | | | **Note:** See the memory chapter for details about which memories are available for EasyDMA. |

## 6.14.15.48 FREQUENCY

Address offset: 0x508

Frequency

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | B A A A A A A A |
| **Reset 0x00000002** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | FREQUENCY | | [0..100] | Radio channel frequency<br><br>Frequency = 2400 + FREQUENCY (MHz). |
| B | RW | MAP | | | Channel map selection. |
| | | | Default | 0 | Channel map between 2400 MHZ .. 2500 MHz<br><br>Frequency = 2400 + FREQUENCY (MHz) |
| | | | Low | 1 | Channel map between 2360 MHZ .. 2460 MHz<br><br>Frequency = 2360 + FREQUENCY (MHz) |

## 6.14.15.49 TXPOWER

Address offset: 0x50C

Output power

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description | |
|---|---|---|---|---|---|---|
| A | RW | TXPOWER | | | RADIO output power<br><br>Output power in number of dBm, i.e. if the value -20 is specified the output power will be set to -20 dBm. | |
| | | | Pos4dBm | 0x4 | +4 dBm | |
| | | | Pos3dBm | 0x3 | +3 dBm | |
| | | | 0dBm | 0x0 | 0 dBm | |
| | | | Neg4dBm | 0xFC | -4 dBm | |
| | | | Neg8dBm | 0xF8 | -8 dBm | |
| | | | Neg12dBm | 0xF4 | -12 dBm | |
| | | | Neg16dBm | 0xF0 | -16 dBm | |
| | | | Neg20dBm | 0xEC | -20 dBm | |
| | | | Neg30dBm | 0xE2 | -40 dBm | Deprecated |
| | | | Neg40dBm | 0xD8 | -40 dBm | |

## 6.14.15.50 MODE

Address offset: 0x510

Data rate and modulation

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | MODE | | | Radio data rate and modulation setting. The radio supports frequency-shift keying (FSK) modulation. |
| | | | Nrf_1Mbit | 0 | 1 Mbps Nordic proprietary radio mode |
| | | | Nrf_2Mbit | 1 | 2 Mbps Nordic proprietary radio mode |
| | | | Ble_1Mbit | 3 | 1 Mbps BLE |
| | | | Ble_2Mbit | 4 | 2 Mbps BLE |
| | | | Ble_LR125Kbit | 5 | Long range 125 kbps TX, 125 kbps and 500 kbps RX |
| | | | Ble_LR500Kbit | 6 | Long range 500 kbps TX, 125 kbps and 500 kbps RX |
| | | | Ieee802154_250Kbit | 15 | IEEE 802.15.4-2006 250 kbps |

## 6.14.15.51 PCNF0

Address offset: 0x514

Packet configuration register 0

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | J J I H H G G F E E E E C A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | LFLEN | | | Length on air of LENGTH field in number of bits. |
| C | RW | S0LEN | | | Length on air of S0 field in number of bytes. |
| E | RW | S1LEN | | | Length on air of S1 field in number of bits. |
| F | RW | S1INCL | | | Include or exclude S1 field in RAM |
| | | | Automatic | 0 | Include S1 field in RAM only if S1LEN > 0 |
| | | | Include | 1 | Always include S1 field in RAM independent of S1LEN |
| G | RW | CILEN | | | Length of code indicator - long range |
| H | RW | PLEN | | | Length of preamble on air. Decision point: TASKS_START task |
| | | | 8bit | 0 | 8-bit preamble |
| | | | 16bit | 1 | 16-bit preamble |
| | | | 32bitZero | 2 | 32-bit zero preamble - used for IEEE 802.15.4 |
| | | | LongRange | 3 | Preamble - used for BLE long range |
| I | RW | CRCINC | | | Indicates if LENGTH field contains CRC or not |
| | | | Exclude | 0 | LENGTH does not contain CRC |
| | | | Include | 1 | LENGTH includes CRC |
| J | RW | TERMLEN | | | Length of TERM field in Long Range operation |

## 6.14.15.52 PCNF1

Address offset: 0x518

Packet configuration register 1

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | E D C C C B B B B B B B A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | MAXLEN | | [0..255] | Maximum length of packet payload. If the packet payload is larger than MAXLEN, the radio will truncate the payload to MAXLEN. |
| B | RW | STATLEN | | [0..255] | Static length in number of bytes<br>The static length parameter is added to the total length of the payload when sending and receiving packets, e.g. if the static length is set to N the radio will receive or send N bytes more than what is defined in the LENGTH field of the packet. |
| C | RW | BALEN | | [2..4] | Base address length in number of bytes<br>The address field is composed of the base address and the one byte long address prefix, e.g. set BALEN=2 to get a total address of 3 bytes. |
| D | RW | ENDIAN | | | On-air endianness of packet, this applies to the S0, LENGTH, S1, and the PAYLOAD fields. |
| | | | Little | 0 | Least significant bit on air first |
| | | | Big | 1 | Most significant bit on air first |
| E | RW | WHITEEN | | | Enable or disable packet whitening |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |

## 6.14.15.53 BASE0

Address offset: 0x51C

Base address 0

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | BASE0 | | | Base address 0 |

## 6.14.15.54 BASE1

Address offset: 0x520

Base address 1

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | BASE1 | | | Base address 1 |

## 6.14.15.55 PREFIX0

Address offset: 0x524

Prefixes bytes for logical addresses 0-3

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | D D D D D D D D C C C C C C C C B B B B B B B B A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-D | RW | AP[i] (i=0..3) | | | Address prefix i. |

## 6.14.15.56 PREFIX1

Address offset: 0x528

Prefixes bytes for logical addresses 4-7

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | D D D D D D D D C C C C C C C C B B B B B B B B A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-D | RW | AP[i] (i=4..7) | | | Address prefix i. |

## 6.14.15.57 TXADDRESS

Address offset: 0x52C

Transmit address select

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | TXADDRESS | | | Transmit address select<br><br>Logical address to be used when transmitting a packet |

## 6.14.15.58 RXADDRESSES

Address offset: 0x530

Receive address select

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-H | RW | ADDR[i] (i=0..7) | | | Enable or disable reception on logical address i. |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |

## 6.14.15.59 CRCCNF

Address offset: 0x534

CRC configuration

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | B B A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | LEN | | [1..3] | CRC length in number of bytes.<br><br>**Note:** For MODE Ble_LR125Kbit and Ble_LR500Kbit, only LEN set to 3 is supported |
| | | | Disabled | 0 | CRC length is zero and CRC calculation is disabled |
| | | | One | 1 | CRC length is one byte and CRC calculation is enabled |
| | | | Two | 2 | CRC length is two bytes and CRC calculation is enabled |
| | | | Three | 3 | CRC length is three bytes and CRC calculation is enabled |
| B | RW | SKIPADDR | | | Include or exclude packet address field out of CRC calculation. |
| | | | Include | 0 | CRC calculation includes address field |
| | | | Skip | 1 | CRC calculation does not include address field. The CRC calculation will start at the first byte after the address. |
| | | | Ieee802154 | 2 | CRC calculation as per 802.15.4 standard. Starting at first byte after length field. |

## 6.14.15.60 CRCPOLY

Address offset: 0x538

CRC polynomial

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | CRCPOLY | | | CRC polynomial<br><br>Each term in the CRC polynomial is mapped to a bit in this register which index corresponds to the term's exponent. The least significant term/bit is hardwired internally to 1, and bit number 0 of the register content is ignored by the hardware. The following example is for an 8 bit CRC polynomial: x8 + x7 + x3 + x2 + 1 = 1 1000 1101 . |

## 6.14.15.61 CRCINIT

Address offset: 0x53C

CRC initial value

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | CRCINIT | | | CRC initial value<br><br>Initial value for CRC calculation |

## 6.14.15.62 TIFS

Address offset: 0x544

Interframe spacing in µs

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | TIFS | | | Interframe spacing in µs<br><br>Interframe space is the time interval between two consecutive packets. It is defined as the time, in microseconds, from the end of the last bit of the previous packet to the start of the first bit of the subsequent packet. |

## 6.14.15.63 RSSISAMPLE

Address offset: 0x548

RSSI sample

| Bit number | 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 |
|---|---|
| ID | A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | RSSISAMPLE | | [0..127] | RSSI sample<br><br>RSSI sample result. The value of this register is read as a positive value while the actual received signal strength is a negative value. Actual received signal strength is therefore as follows: received signal strength = -A dBm |

## 6.14.15.64 STATE

Address offset: 0x550

Current radio state

| Bit number | 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 |
|---|---|
| ID | A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | STATE | | | Current radio state |
| | | | Disabled | 0 | RADIO is in the Disabled state |
| | | | RxRu | 1 | RADIO is in the RXRU state |
| | | | RxIdle | 2 | RADIO is in the RXIDLE state |
| | | | Rx | 3 | RADIO is in the RX state |
| | | | RxDisable | 4 | RADIO is in the RXDISABLED state |
| | | | TxRu | 9 | RADIO is in the TXRU state |
| | | | TxIdle | 10 | RADIO is in the TXIDLE state |
| | | | Tx | 11 | RADIO is in the TX state |
| | | | TxDisable | 12 | RADIO is in the TXDISABLED state |

## 6.14.15.65 DATAWHITEIV

Address offset: 0x554

Data whitening initial value

| Bit number | 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 |
|---|---|
| ID | A A A A A A |
| **Reset 0x00000040** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | DATAWHITEIV | | | Data whitening initial value. Bit 6 is hardwired to '1', writing '0' to it has no effect, and it will always be read back and used by the device as '1'.<br>Bit 0 corresponds to Position 6 of the LSFR, Bit 1 to Position 5, etc. |

## 6.14.15.66 BCC

Address offset: 0x560

Bit counter compare

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | BCC | | | Bit counter compare<br>Bit counter compare register |

## 6.14.15.67 DAB[n] (n=0..7)

Address offset: 0x600 + (n × 0x4)

Device address base segment n

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | DAB | | | Device address base segment n |

## 6.14.15.68 DAP[n] (n=0..7)

Address offset: 0x620 + (n × 0x4)

Device address prefix n

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | DAP | | | Device address prefix n |

## 6.14.15.69 DACNF

Address offset: 0x640

Device address match configuration

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-H | RW | ENA[i] (i=0..7) | | | Enable or disable device address matching using device address i |
| | | | Disabled | 0 | Disabled |
| | | | Enabled | 1 | Enabled |
| I-P | RW | TXADD[i] (i=0..7) | | | TxAdd for device address i |

## 6.14.15.70 MHRMATCHCONF

Address offset: 0x644

Search pattern configuration

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | MHRMATCHCONF | | | Search pattern configuration |

## 6.14.15.71 MHRMATCHMAS

Address offset: 0x648

Pattern mask

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | MHRMATCHMAS | | | Pattern mask |

## 6.14.15.72 MODECNF0

Address offset: 0x650

Radio mode configuration register 0

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C C A |
| **Reset 0x00000200** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | RU | | | Radio ramp-up time |
| | | | Default | 0 | Default ramp-up time (tRXEN and tTXEN), compatible with firmware written for nRF51 |
| | | | Fast | 1 | Fast ramp-up (tRXEN,FAST and tTXEN,FAST), see electrical specification for more information<br><br>When enabled, TIFS is not enforced by hardware and software needs to control when to turn on the Radio. |

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | C C A |
| **Reset 0x00000200** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| C | RW | DTX | | | Default TX value<br>Specifies what the RADIO will transmit when it is not started, i.e. between:<br>RADIO.EVENTS_READY and RADIO.TASKS_START<br>RADIO.EVENTS_END and RADIO.TASKS_START<br>RADIO.EVENTS_END and RADIO.EVENTS_DISABLED<br>**Note:** For 802.15.4 and BLE LR mode, only Center is a valid setting |
| | | | B1 | 0 | Transmit '1' |
| | | | B0 | 1 | Transmit '0' |
| | | | Center | 2 | Transmit center frequency<br>When tuning the crystal for center frequency, the RADIO must be set in DTX = Center mode to be able to achieve the expected accuracy |

## 6.14.15.73 SFD

Address offset: 0x660

IEEE 802.15.4 start of frame delimiter

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A |
| **Reset 0x000000A7** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 1 1 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | SFD | | | IEEE 802.15.4 start of frame delimiter |

## 6.14.15.74 EDCNT

Address offset: 0x664

IEEE 802.15.4 energy detect loop count

Number of iterations to perform an ED scan. If set to 0 one scan is performed, otherwise the specified number + 1 of ED scans will be performed and the max ED value tracked in EDSAMPLE.

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | EDCNT | | | IEEE 802.15.4 energy detect loop count |

## 6.14.15.75 EDSAMPLE

Address offset: 0x668

IEEE 802.15.4 energy detect level

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EDLVL | | [0..127] | IEEE 802.15.4 energy detect level<br>Register value must be converted to IEEE 802.15.4 range by an 8-bit saturating multiplication by factor ED_RSSISCALE, as shown in the code example for ED sampling |

## 6.14.15.76 CCACTRL

Address offset: 0x66C

IEEE 802.15.4 clear channel assessment control

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | D D D D D D D D C C C C C C C C B B B B B B B B A A A |
| **Reset 0x052D0000** | | | | **0 0 0 0 0 1 0 1 0 0 1 0 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | CCAMODE | | | CCA mode of operation |
| | | | EdMode | 0 | Energy above threshold<br>Will report busy whenever energy is detected above CCAEDTHRES |
| | | | CarrierMode | 1 | Carrier seen<br>Will report busy whenever compliant IEEE 802.15.4 signal is seen |
| | | | CarrierAndEdMode | 2 | Energy above threshold AND carrier seen |
| | | | CarrierOrEdMode | 3 | Energy above threshold OR carrier seen |
| | | | EdModeTest1 | 4 | Energy above threshold test mode that will abort when first ED measurement over threshold is seen. No averaging. |
| B | RW | CCAEDTHRES | | | CCA energy busy threshold. Used in all the CCA modes except CarrierMode.<br>Must be converted from IEEE 802.15.4 range by dividing by factor ED_RSSISCALE - similar to EDSAMPLE register |
| C | RW | CCACORRTHRES | | | CCA correlator busy threshold. Only relevant to CarrierMode, CarrierAndEdMode, and CarrierOrEdMode. |
| D | RW | CCACORRCNT | | | Limit for occurances above CCACORRTHRES. When not equal to zero the corrolator based signal detect is enabled. |

## 6.14.15.77 DFEMODE

Address offset: 0x900

Whether to use Angle-of-Arrival (AOA) or Angle-of-Departure (AOD)

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | DFEOPMODE | | | Direction finding operation mode |
| | | | Disabled | 0 | Direction finding mode disabled |
| | | | AoD | 2 | Direction finding mode set to AoD |
| | | | AoA | 3 | Direction finding mode set to AoA |

## 6.14.15.78 CTEINLINECONF

Address offset: 0x904

Configuration for CTE inline mode

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | I I I I I I I I H H H H H H H H G G G F F F E E C B A |
| **Reset 0x00002800** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | CTEINLINECTRLEN | | | Enable parsing of CTEInfo from received packet in BLE modes |
| | | | Enabled | 1 | Parsing of CTEInfo is enabled |
| | | | Disabled | 0 | Parsing of CTEInfo is disabled |
| B | RW | CTEINFOINS1 | | | CTEInfo is S1 byte or not |
| | | | InS1 | 1 | CTEInfo is in S1 byte (data PDU) |
| | | | NotInS1 | 0 | CTEInfo is NOT in S1 byte (advertising PDU) |
| C | RW | CTEERRORHANDLING | | | Sampling/switching if CRC is not OK |
| | | | Yes | 1 | Sampling and antenna switching also when CRC is not OK |
| | | | No | 0 | No sampling and antenna switching when CRC is not OK |
| E | RW | CTETIMEVALIDRANGE | | | Max range of CTETime<br>**Note:** Valid range is 2-20 in BLE core spec. If larger than 20, it can be an indication of an error in the received packet. |
| | | | 20 | 0 | 20 in 8us unit (default)<br>Set to 20 if parsed CTETime is larger han 20 |
| | | | 31 | 1 | 31 in 8us unit |
| | | | 63 | 2 | 63 in 8us unit |
| F | RW | CTEINLINERXMODE1US | | | Spacing between samples for the samples in the SWITCHING period when CTEINLINEMODE is set<br>When the device is in AoD mode, this is used when the received CTEType is "AoD 1 us". When in AoA mode, this is used when TSWITCHSPACING is 2 us. |
| | | | 4us | 1 | 4us |
| | | | 2us | 2 | 2us |
| | | | 1us | 3 | 1us |
| | | | 500ns | 4 | 0.5us |
| | | | 250ns | 5 | 0.25us |
| | | | 125ns | 6 | 0.125us |
| G | RW | CTEINLINERXMODE2US | | | Spacing between samples for the samples in the SWITCHING period when CTEINLINEMODE is set<br>When the device is in AoD mode, this is used when the received CTEType is "AoD 2 us". When in AoA mode, this is used when TSWITCHSPACING is 4 us. |
| | | | 4us | 1 | 4us |
| | | | 2us | 2 | 2us |
| | | | 1us | 3 | 1us |
| | | | 500ns | 4 | 0.5us |
| | | | 250ns | 5 | 0.25us |
| | | | 125ns | 6 | 0.125us |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | I I I I I I I I H H H H H H H H G G G F F F E E C B A |
| **Reset 0x00002800** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| H | RW | S0CONF | | | S0 bit pattern to match<br>The least significant bit always corresponds to the first bit of S0 received. |
| I | RW | S0MASK | | | S0 bit mask to set which bit to match<br>The least significant bit always corresponds to the first bit of S0 received. |

## 6.14.15.79 DFECTRL1

Address offset: 0x910

Various configuration for Direction finding

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | I I I I G G G F E E E C C C B A A A A A A |
| **Reset 0x00023282** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 1 0 0 1 0 1 0 0 0 0 0 1 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | NUMBEROF8US | | | Length of the AoA/AoD procedure in number of 8 us units<br>Always used in TX mode, but in RX mode only when CTEINLINECTRLEN is 0 |
| B | RW | DFEINEXTENSION | | | Add CTE extension and do antenna switching/sampling in this extension |
| | | | CRC | 1 | AoA/AoD procedure triggered at end of CRC |
| | | | Payload | 0 | Antenna switching/sampling is done in the packet payload |
| C | RW | TSWITCHSPACING | | | Interval between every time the antenna is changed in the SWITCHING state |
| | | | 4us | 1 | 4us |
| | | | 2us | 2 | 2us |
| | | | 1us | 3 | 1us |
| E | RW | TSAMPLESPACINGREF | | | Interval between samples in the REFERENCE period |
| | | | 4us | 1 | 4us |
| | | | 2us | 2 | 2us |
| | | | 1us | 3 | 1us |
| | | | 500ns | 4 | 0.5us |
| | | | 250ns | 5 | 0.25us |
| | | | 125ns | 6 | 0.125us |
| F | RW | SAMPLETYPE | | | Whether to sample I/Q or magnitude/phase |
| | | | IQ | 0 | Complex samples in I and Q |
| | | | MagPhase | 1 | Complex samples as magnitude and phase |
| G | RW | TSAMPLESPACING | | | Interval between samples in the SWITCHING period when CTEINLINECTRLEN is 0<br>**Note:** Not used when CTEINLINECTRLEN is set. Then either CTEINLINERXMODE1US or CTEINLINERXMODE2US are used. |
| | | | 4us | 1 | 4us |
| | | | 2us | 2 | 2us |
| | | | 1us | 3 | 1us |
| | | | 500ns | 4 | 0.5us |

| Bit number | 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 |
|---|---|
| ID | I I I I G G G F E E E C C C B A A A A A A |
| **Reset 0x00023282** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 1 0 0 1 0 1 0 0 0 0 0 1 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| | | | 250ns | 5 | 0.25us |
| | | | 125ns | 6 | 0.125us |
| I | RW | AGCBACKOFFGAIN | | | Gain will be lowered by the specified number of gain steps at the start of CTE |

**Note:** First LNAGAIN gain drops, then MIXGAIN, then AAFGAIN

## 6.14.15.80 DFECTRL2

Address offset: 0x914

Start offset for Direction finding

| Bit number | 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 |
|---|---|
| ID | B B B B B B B B B B B B B A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | TSWITCHOFFSET | | | Signed value offset after the end of the CRC before starting switching in number of 16M cycles |
| B | RW | TSAMPLEOFFSET | | | Signed value offset before starting sampling in number of 16M cycles relative to the beginning of the REFERENCE state - 12 us after switching start |

## 6.14.15.81 SWITCHPATTERN

Address offset: 0x928

GPIO patterns to be used for each antenna

Maximum 8 GPIOs can be controlled. To secure correct signal levels on the pins, the pins must be configured in the GPIO peripheral as described in Pin configuration.

If, during switching, the total number of antenna slots is bigger than the number of written patterns, the RADIO loops back to the pattern used after the reference pattern.

A minimum number of 3 patterns must be written.

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | SWITCHPATTERN | | | Fill array of GPIO patterns for antenna control<br>The GPIO pattern array size is 40 entries.<br>When written, bit n corresponds to the GPIO configured in PSEL.DFEGPIO[n].<br>When read, returns the number of GPIO patterns written since the last time the array was cleared. Use CLEARPATTERN to clear the array. |

## 6.14.15.82 CLEARPATTERN

Address offset: 0x92C

Clear the GPIO pattern array for antenna control

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | CLEARPATTERN | | | Clears GPIO pattern array for antenna control |
| | | | Clear | 1 | Clear the GPIO pattern |

## 6.14.15.83 PSEL.DFEGPIO[n] (n=0..7)

Address offset: 0x930 + (n × 0x4)

Pin select for DFE pin n

Must be set before enabling the radio

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| B | RW | PORT | | [0..1] | Port number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.14.15.84 DFEPACKET.PTR

Address offset: 0x950

Data pointer

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PTR | | | Data pointer |

**Note:** See the memory chapter for details about which memories are available for EasyDMA.

## 6.14.15.85 DFEPACKET.MAXCNT

Address offset: 0x954

Maximum number of buffer words to transfer

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00001000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | MAXCNT | | | Maximum number of buffer words to transfer |

## 6.14.15.86 DFEPACKET.AMOUNT

Address offset: 0x958

Number of samples transferred in the last transaction

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | AMOUNT | | | Number of samples transferred in the last transaction |

## 6.14.15.87 POWER

Address offset: 0xFFC

Peripheral power control

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000001** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | POWER | | | Peripheral power control. The peripheral and its registers will be reset to its initial state by switching the peripheral off and then back on again. |
| | | | Disabled | 0 | Peripheral is powered off |
| | | | Enabled | 1 | Peripheral is powered on |

# 6.14.16 Electrical specification

## 6.14.16.1 General radio characteristics

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $f_{OP}$ | Operating frequencies | 2360 | | 2500 | MHz |
| $f_{PLL,CH,SP}$ | PLL channel spacing | | 1 | | MHz |
| $f_{DELTA,1M}$ | Frequency deviation @ 1 Mbps | | ±170 | | kHz |
| $f_{DELTA,BLE,1M}$ | Frequency deviation @ BLE 1 Mbps | | ±250 | | kHz |
| $f_{DELTA,2M}$ | Frequency deviation @ 2 Mbps | | ±320 | | kHz |
| $f_{DELTA,BLE,2M}$ | Frequency deviation @ BLE 2 Mbps | | ±500 | | kHz |
| $fsk_{BPS}$ | On-the-air data rate | 125 | | 2000 | kbps |
| $f_{chip,\ IEEE\ 802.15.4}$ | Chip rate in IEEE 802.15.4 mode | | 2000 | | kchip/s |

## 6.14.16.2 Radio current consumption (transmitter)

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $I_{TX,PLUS4dBM,DCDC}$ | TX only run current (DC/DC, 3 V) $P_{RF}$ = +4 dBm | | 7.0 | | mA |

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $I_{TX,PLUS4dBM}$ | TX only run current $P_{RF}$ = +4 dBm | | 15.4 | | mA |
| $I_{TX,0dBM,DCDC}$ | TX only run current (DC/DC, 3 V)$P_{RF}$ = 0 dBm | | 4.6 | | mA |
| $I_{TX,0dBM}$ | TX only run current $P_{RF}$ = 0 dBm | | 10.1 | | mA |
| $I_{TX,MINUS4dBM,DCDC}$ | TX only run current DC/DC, 3 V $P_{RF}$ = -4 dBm | | 3.6 | | mA |
| $I_{TX,MINUS4dBM}$ | TX only run current $P_{RF}$ = -4 dBm | | 7.8 | | mA |
| $I_{TX,MINUS8dBM,DCDC}$ | TX only run current DC/DC, 3 V $P_{RF}$ = -8 dBm | | 3.2 | | mA |
| $I_{TX,MINUS8dBM}$ | TX only run current $P_{RF}$ = -8 dBm | | 6.8 | | mA |
| $I_{TX,MINUS12dBM,DCDC}$ | TX only run current DC/DC, 3 V $P_{RF}$ = -12 dBm | | 2.9 | | mA |
| $I_{TX,MINUS12dBM}$ | TX only run current $P_{RF}$ = -12 dBm | | 6.2 | | mA |
| $I_{TX,MINUS16dBM,DCDC}$ | TX only run current DC/DC, 3 V $P_{RF}$ = -16 dBm | | 2.7 | | mA |
| $I_{TX,MINUS16dBM}$ | TX only run current $P_{RF}$ = -16 dBm | | 5.7 | | mA |
| $I_{TX,MINUS20dBM,DCDC}$ | TX only run current DC/DC, 3 V $P_{RF}$ = -20 dBm | | 2.5 | | mA |
| $I_{TX,MINUS20dBM}$ | TX only run current $P_{RF}$ = -20 dBm | | 5.4 | | mA |
| $I_{TX,MINUS40dBM,DCDC}$ | TX only run current DC/DC, 3 V $P_{RF}$ = -40 dBm | | 2.1 | | mA |
| $I_{TX,MINUS40dBM}$ | TX only run current $P_{RF}$ = -40 dBm | | 4.3 | | mA |
| $I_{START,TX,DCDC}$ | TX start-up current DC/DC, 3 V, $P_{RF}$ = 4 dBm | .. | .. | .. | mA |
| $I_{START,TX}$ | TX start-up current, $P_{RF}$ = 4 dBm | .. | .. | .. | mA |

## 6.14.16.3 Radio current consumption (Receiver)

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $I_{RX,1M,DCDC}$ | RX only run current (DC/DC, 3 V) 1 Mbps/1 Mbps BLE | | 4.6 | | mA |
| $I_{RX,1M}$ | RX only run current (LDO, 3 V) 1 Mbps/1 Mbps BLE | | 10.0 | | mA |
| $I_{RX,2M,DCDC}$ | RX only run current (DC/DC, 3 V) 2 Mbps/2 Mbps BLE | | 5.2 | | mA |
| $I_{RX,2M}$ | RX only run current (LDO, 3 V) 2 Mbps/2 Mbps BLE | | 11.2 | | mA |
| $I_{START,RX,1M,DCDC}$ | RX start-up current (DC/DC, 3 V) 1 Mbps/1 Mbps BLE | | 3.5 | | mA |
| $I_{START,RX,1M}$ | RX start-up current 1 Mbps/1 Mbps BLE | | 6.7 | | mA |

## 6.14.16.4 Transmitter specification

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $P_{RF}$ | Maximum output power | | 4.0 | | dBm |
| $P_{RFC}$ | RF power control range | | 24 | | dB |
| $P_{RFCR}$ | RF power accuracy | | | ±4 | dB |
| $P_{RF1,1}$ | 1st Adjacent Channel Transmit Power 1 MHz (1 Mbps) | | -25 | | dBc |
| $P_{RF2,1}$ | 2nd Adjacent Channel Transmit Power 2 MHz (1 Mbps) | | -50 | | dBc |
| $P_{RF1,2}$ | 1st Adjacent Channel Transmit Power 2 MHz (2 Mbps) | | -25 | | dBc |
| $P_{RF2,2}$ | 2nd Adjacent Channel Transmit Power 4 MHz (2 Mbps) | | -50 | | dBc |
| $E_{vm}$ | Error vector magnitude IEEE 802.15.4 | | 12 | | %rms |
| $P_{harm2nd,\ IEEE\ 802.15.4}$ | 2nd harmonics in IEEE 802.15.4 mode | | -45 | | dBm |
| $P_{harm3rd,\ IEEE\ 802.15.4}$ | 3rd harmonics in IEEE 802.15.4 | | | | dBm |

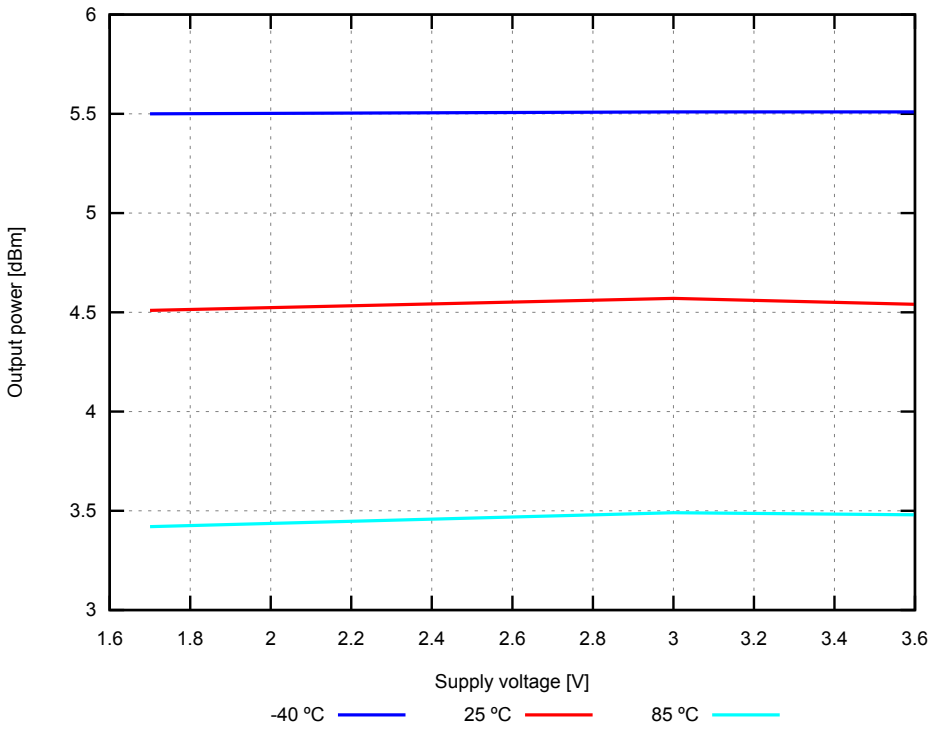



*Figure 85: Output power, 1 Mbps Bluetooth low energy mode, at maximum TXPOWER setting (typical values)*

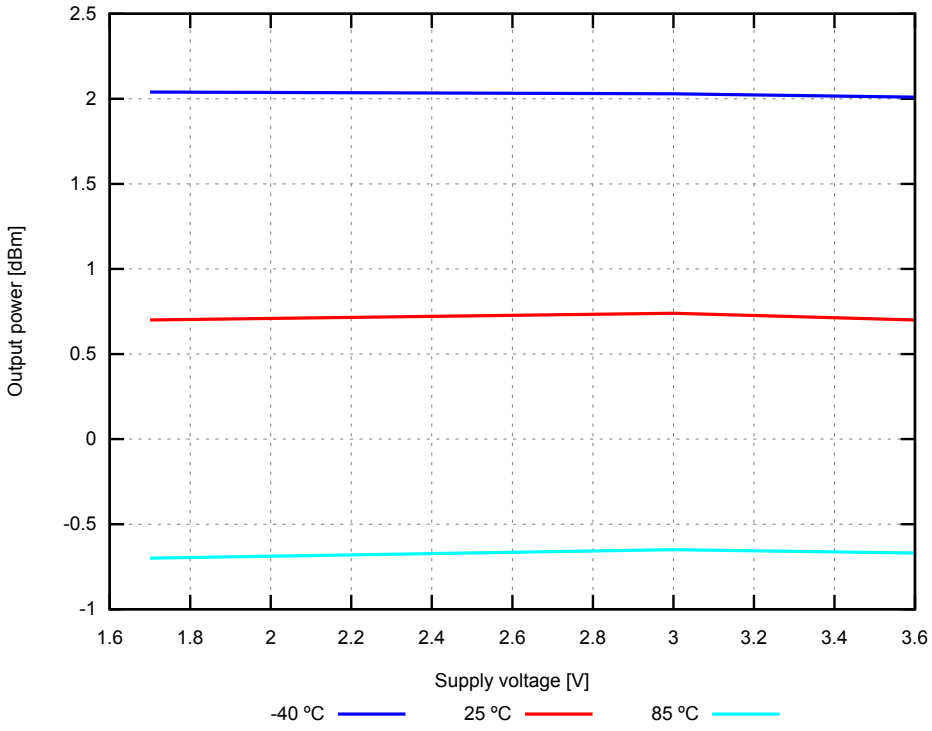



*Figure 86: Output power, 1 Mbps Bluetooth low energy mode, at 0 dBm TXPOWER setting (typical values)*

## 6.14.16.5 Receiver operation

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $P_{RX,MAX}$ | Maximum received signal strength at < 0.1% PER | | 0 | | dBm |
| $P_{SENS,IT,1M}$ | Sensitivity, 1 Mbps nRF mode ideal transmitter[14] | | -94 | | dBm |
| $P_{SENS,IT,2M}$ | Sensitivity, 2 Mbps nRF mode ideal transmitter[15] | | -91 | | dBm |
| $P_{SENS,IT,SP,1M,BLE}$ | Sensitivity, 1 Mbps BLE ideal transmitter, packet length ≤ 37 bytes BER=1E-3[16] | | -97 | | dBm |
| $P_{SENS,IT,LP,1M,BLE}$ | Sensitivity, 1 Mbps BLE ideal transmitter, packet length ≥ 128 bytes BER=1E-4 [17] | | -96 | | dBm |
| $P_{SENS,IT,SP,2M,BLE}$ | Sensitivity, 2 Mbps BLE ideal transmitter, packet length ≤ 37 bytes | | -94 | | dBm |
| $P_{SENS,IT,BLE\ LE125k}$ | Sensitivity, 125 kbps BLE mode | | -104 | | dBm |
| $P_{SENS,IT,BLE\ LE500k}$ | Sensitivity, 500 kbps BLE mode | | -100 | | dBm |
| $P_{SENS,IEEE\ 802.15.4}$ | Sensitivity in IEEE 802.15.4 mode | | -101 | | dBm |

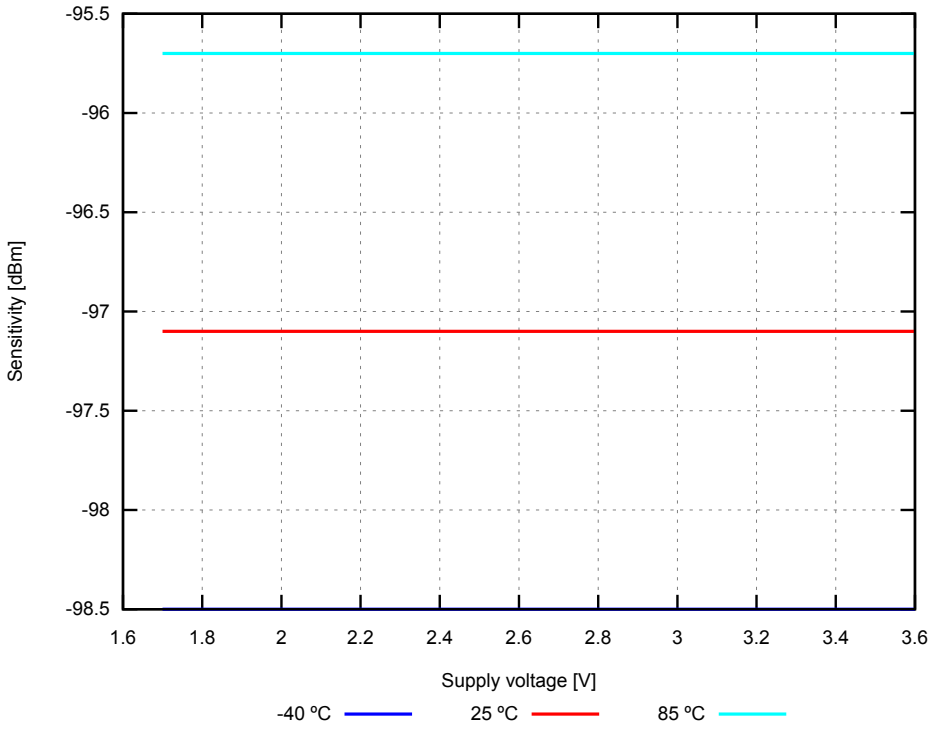



*Figure 87: Sensitivity, 1 Mbps Bluetooth low energy mode, Regulator = LDO (typical values)*

## 6.14.16.6 RX selectivity

RX selectivity with equal modulation on interfering signal[18]

[14] Typical sensitivity applies when ADDR0 is used for receiver address correlation. When ADDR[1...7] are used for receiver address correlation, the typical sensitivity for this mode is degraded by 3 dB.

[15] Typical sensitivity applies when ADDR0 is used for receiver address correlation. When ADDR[1..7] are used for receiver address correlation, the typical sensitivity for this mode is degraded by 3 dB.

[16] As defined in the *Bluetooth Core Specification v4.0 Volume 6: Core System Package (Low Energy Controller Volume)*

[17] Equivalent BER limit < 10E-04

[18] Desired signal level at PIN = -67 dBm. One interferer is used, having equal modulation as the desired signal. The input power of the interferer where the sensitivity equals BER = 0.1% is presented

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $C/I_{1M,co\text{-}channel}$ | 1Mbps mode, Co-Channel interference | | 9 | | dB |
| $C/I_{1M,-1MHz}$ | 1 Mbps mode, Adjacent (-1 MHz) interference | | -2 | | dB |
| $C/I_{1M,+1MHz}$ | 1 Mbps mode, Adjacent (+1 MHz) interference | | -10 | | dB |
| $C/I_{1M,-2MHz}$ | 1 Mbps mode, Adjacent (-2 MHz) interference | | -19 | | dB |
| $C/I_{1M,+2MHz}$ | 1 Mbps mode, Adjacent (+2 MHz) interference | | -42 | | dB |
| $C/I_{1M,-3MHz}$ | 1 Mbps mode, Adjacent (-3 MHz) interference | | -38 | | dB |
| $C/I_{1M,+3MHz}$ | 1 Mbps mode, Adjacent (+3 MHz) interference | | -48 | | dB |
| $C/I_{1M,\pm6MHz}$ | 1 Mbps mode, Adjacent (≥6 MHz) interference | | -50 | | dB |
| $C/I_{1MBLE,co\text{-}channel}$ | 1 Mbps BLE mode, Co-Channel interference | | 6 | | dB |
| $C/I_{1MBLE,-1MHz}$ | 1 Mbps BLE mode, Adjacent (-1 MHz) interference | | -2 | | dB |
| $C/I_{1MBLE,+1MHz}$ | 1 Mbps BLE mode, Adjacent (+1 MHz) interference | | -9 | | dB |
| $C/I_{1MBLE,-2MHz}$ | 1 Mbps BLE mode, Adjacent (-2 MHz) interference | | -22 | | dB |
| $C/I_{1MBLE,+2MHz}$ | 1 Mbps BLE mode, Adjacent (+2 MHz) interference | | -46 | | dB |
| $C/I_{1MBLE,>3MHz}$ | 1 Mbps BLE mode, Adjacent (≥3 MHz) interference | | -50 | | dB |
| $C/I_{1MBLE,image}$ | Image frequency interference | | -22 | | dB |
| $C/I_{1MBLE,image,1MHz}$ | Adjacent (1 MHz) interference to in-band image frequency | | -35 | | dB |
| $C/I_{2M,co\text{-}channel}$ | 2 Mbps mode, Co-Channel interference | | 10 | | dB |
| $C/I_{2M,-2MHz}$ | 2 Mbps mode, Adjacent (-2 MHz) interference | | 6 | | dB |
| $C/I_{2M,+2MHz}$ | 2 Mbps mode, Adjacent (+2 MHz) interference | | -14 | | dB |
| $C/I_{2M,-4MHz}$ | 2 Mbps mode, Adjacent (-4 MHz) interference | | -20 | | dB |
| $C/I_{2M,+4MHz}$ | 2 Mbps mode, Adjacent (+4 MHz) interference | | -44 | | dB |
| $C/I_{2M,-6MHz}$ | 2 Mbps mode, Adjacent (-6 MHz) interference | | -42 | | dB |
| $C/I_{2M,+6MHz}$ | 2 Mbps mode, Adjacent (+6 MHz) interference | | -47 | | dB |
| $C/I_{2M,\geq12MHz}$ | 2 Mbps mode, Adjacent (≥12 MHz) interference | | -52 | | dB |
| $C/I_{2MBLE,co\text{-}channel}$ | 2 Mbps BLE mode, Co-Channel interference | | 6 | | dB |
| $C/I_{2MBLE,\pm2MHz}$ | 2 Mbps BLE mode, Adjacent (±2 MHz) interference | | -2 | | dB |
| $C/I_{2MBLE,\pm4MHz}$ | 2 Mbps BLE mode, Adjacent (±4 MHz) interference | | -48 | | dB |
| $C/I_{2MBLE,\geq6MHz}$ | 2 Mbps BLE mode, Adjacent (≥6 MHz) interference | | -50 | | dB |
| $C/I_{2MBLE,image}$ | Image frequency interference | | -29 | | dB |
| $C/I_{2MBLE,image,\ 2MHz}$ | Adjacent (2 MHz) interference to in-band image frequency | | -44 | | dB |
| $C/I_{125k\ BLE\ LR,co\text{-}channel}$ | 125 kbps BLE LR mode, Co-Channel interference | | 4 | | dB |
| $C/I_{125k\ BLE\ LR,-1MHz}$ | 125 kbps BLE LR mode, Adjacent (-1 MHz) interference | | -9 | | dB |
| $C/I_{125k\ BLE\ LR,+1MHz}$ | 125 kbps BLE LR mode, Adjacent (+1 MHz) interference | | -16 | | dB |
| $C/I_{125k\ BLE\ LR,-2MHz}$ | 125 kbps BLE LR mode, Adjacent (-2 MHz) interference | | -30 | | dB |
| $C/I_{125k\ BLE\ LR,+2MHz}$ | 125 kbps BLE LR mode, Adjacent (+2 MHz) interference | | -50 | | dB |
| $C/I_{125k\ BLE\ LR,>3MHz}$ | 125 kbps BLE LR mode, Adjacent (≥3 MHz) interference | | -55 | | dB |
| $C/I_{125k\ BLE\ LR,image}$ | Image frequency interference | | -30 | | dB |
| $C/I_{IEEE\ 802.15.4,-5MHz}$ | IEEE 802.15.4 mode, Adjacent (-5 MHz) rejection | | 33 | | dB |
| $C/I_{IEEE\ 802.15.4,+5MHz}$ | IEEE 802.15.4 mode, Adjacent (+5 MHz) rejection | | 38 | | dB |
| $C/I_{IEEE\ 802.15.4,\|\ 10MHz\|}$ | IEEE 802.15.4 mode, Alternate (10 MHz) rejection | | 48 | | dB |

## 6.14.16.7 RX intermodulation

RX intermodulation[19]

[19] Desired signal level at PIN = -64 dBm. Two interferers with equal input power are used. The interferer closest in frequency is not modulated, the other interferer is modulated equal with the desired signal. The input power of the interferers where the sensitivity equals BER = 0.1% is presented.

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $P_{IMD,5TH,1M}$ | IMD performance, 1 Mbps, 5th offset channel, packet length ≤ 37 bytes | | -33 | | dBm |
| $P_{IMD,5TH,1M,BLE}$ | IMD performance, BLE 1 Mbps, 5th offset channel, packet length ≤ 37 bytes | | -30 | | dBm |
| $P_{IMD,5TH,2M}$ | IMD performance, 2 Mbps, 5th offset channel, packet length ≤ 37 bytes | | -33 | | dBm |
| $P_{IMD,5TH,2M,BLE}$ | IMD performance, BLE 2 Mbps, 5th offset channel, packet length ≤ 37 bytes | | -31 | | dBm |

## 6.14.16.8 Radio timing

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{TXEN,BLE,1M}$ | Time between TXEN task and READY event after channel FREQUENCY configured (1 Mbps BLE and 150 µs TIFS) | 140 | | 140 | µs |
| $t_{TXEN,FAST,BLE,1M}$ | Time between TXEN task and READY event after channel FREQUENCY configured (1 Mbps BLE with fast ramp-up and 150 µs TIFS) | 40 | | 40 | µs |
| $t_{TXDIS,BLE,1M}$ | When in TX, delay between DISABLE task and DISABLED event for MODE = Nrf_1Mbit and MODE = Ble_1Mbit | 6 | | 6 | µs |
| $t_{RXEN,BLE,1M}$ | Time between the RXEN task and READY event after channel FREQUENCY configured (1 Mbps BLE) | 140 | | 140 | µs |
| $t_{RXEN,FAST,BLE,1M}$ | Time between the RXEN task and READY event after channel FREQUENCY configured (1 Mbps BLE with fast ramp-up) | 40 | | 40 | µs |
| $t_{RXDIS,BLE,1M}$ | When in RX, delay between DISABLE task and DISABLED event for MODE = Nrf_1Mbit and MODE = Ble_1Mbit | 0 | | 0 | µs |
| $t_{TXDIS,BLE,2M}$ | When in TX, delay between DISABLE task and DISABLED event for MODE = Nrf_2Mbit and MODE = Ble_2Mbit | 4 | | 4 | µs |
| $t_{RXDIS,BLE,2M}$ | When in RX, delay between DISABLE task and DISABLED event for MODE = Nrf_2Mbit and MODE = Ble_2Mbit | 0 | | 0 | µs |
| $t_{TXEN,IEEE\ 802.15.4}$ | Time between TXEN task and READY event after channel FREQUENCY configured (IEEE 802.15.4) | 130 | | 130 | µs |
| $t_{TXEN,FAST,IEEE\ 802.15.4}$ | Time between TXEN task and READY event after channel FREQUENCY configured (IEEE 802.15.4 with fast ramp-up) | 40 | | 40 | µs |
| $t_{TXDIS,IEEE\ 802.15.4}$ | When in TX, delay between DISABLE task and DISABLED event (IEEE 802.15.4) | 21 | | 21 | µs |
| $t_{RXEN,IEEE\ 802.15.4}$ | Time between the RXEN task and READY event after channel FREQUENCY configured (IEEE 802.15.4) | 130 | | 130 | µs |
| $t_{RXEN,FAST,IEEE\ 802.15.4}$ | Time between the RXEN task and READY event after channel FREQUENCY configured (IEEE 802.15.4 with fast ramp-up) | 40 | | 40 | µs |
| $t_{RXDIS,IEEE\ 802.15.4}$ | When in RX, delay between DISABLE task and DISABLED event (IEEE 802.15.4) | 0.5 | | 0.5 | µs |
| $t_{RX\text{-}to\text{-}TX\ turnaround}$ | Maximum TX-to-RX or RX-to-TX turnaround time in IEEE 802.15.4 mode | | 40 | | µs |

## 6.14.16.9 Received signal strength indicator (RSSI) specifications

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $RSSI_{ACC}$ | RSSI accuracy valid range -90 to -20 dBm | | ±2 | | dB |
| $RSSI_{RESOLUTION}$ | RSSI resolution | | 1 | | dB |
| $RSSI_{PERIOD}$ | RSSI sampling time from RSSI_START task | | 0.25 | | µs |
| $RSSI_{SETTLE}$ | RSSI settling time after signal level change | | 15 | | µs |

### 6.14.16.10 Jitter

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{DISABLEDJITTER}$ | Jitter on DISABLED event relative to END event when shortcut between END and DISABLE is enabled | | 0.25 | | µs |
| $t_{READYJITTER}$ | Jitter on READY event relative to TXEN and RXEN task | | 0.25 | | µs |

### 6.14.16.11 IEEE 802.15.4 energy detection constants

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| ED_RSSISCALE | Scaling value when converting between hardware-reported value and dBm | 4 | 4 | 4 | |
| ED_RSSIOFFS | Offset value when converting between hardware-reported value and dBm | -92 | -92 | -92 | |

## 6.15 RNG — Random number generator

The Random number generator (RNG) generates true non-deterministic random numbers based on internal thermal noise that are suitable for cryptographic purposes. The RNG does not require a seed value.

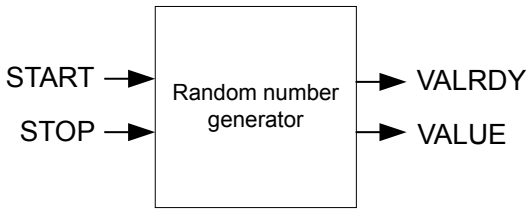



*Figure 88: Random number generator*

The RNG is started by triggering the START task and stopped by triggering the STOP task. When started, new random numbers are generated continuously and written to the VALUE register when ready. A VALRDY event is generated for every new random number that is written to the VALUE register. This means that after a VALRDY event is generated the CPU has the time until the next VALRDY event to read out the random number from the VALUE register before it is overwritten by a new random number.

### 6.15.1 Bias correction

A bias correction algorithm is employed on the internal bit stream to remove any bias toward '1' or '0'. The bits are then queued into an eight-bit register for parallel readout from the VALUE register.

It is possible to enable bias correction in the CONFIG register. This will result in slower value generation, but will ensure a statistically uniform distribution of the random values.

### 6.15.2 Speed

The time needed to generate one random byte of data is unpredictable, and may vary from one byte to the next. This is especially true when bias correction is enabled.

## 6.15.3 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x4000D000 | RNG | RNG | Random number generator | |

*Table 70: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_START | 0x000 | Task starting the random number generator |
| TASKS_STOP | 0x004 | Task stopping the random number generator |
| EVENTS_VALRDY | 0x100 | Event being generated for every new random number written to the VALUE register |
| SHORTS | 0x200 | Shortcuts between local events and tasks |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| CONFIG | 0x504 | Configuration register |
| VALUE | 0x508 | Output random number |

*Table 71: Register overview*

### 6.15.3.1 TASKS_START

Address offset: 0x000

Task starting the random number generator

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_START | | | Task starting the random number generator |
| | | | Trigger | 1 | Trigger task |

### 6.15.3.2 TASKS_STOP

Address offset: 0x004

Task stopping the random number generator

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STOP | | | Task stopping the random number generator |
| | | | Trigger | 1 | Trigger task |

### 6.15.3.3 EVENTS_VALRDY

Address offset: 0x100

Event being generated for every new random number written to the VALUE register

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_VALRDY | | | Event being generated for every new random number written to the VALUE register |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.15.3.4 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | VALRDY_STOP | | | Shortcut between event VALRDY and task STOP |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |

## 6.15.3.5 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | VALRDY | | | Write '1' to enable interrupt for event VALRDY |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.15.3.6 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | VALRDY | | | Write '1' to disable interrupt for event VALRDY |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

### 6.15.3.7 CONFIG

Address offset: 0x504

Configuration register

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | DERCEN | | | Bias correction |
| | | | Disabled | 0 | Disabled |
| | | | Enabled | 1 | Enabled |

### 6.15.3.8 VALUE

Address offset: 0x508

Output random number

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A A A A A A A A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | VALUE | | [0..255] | Generated random number |

## 6.15.4 Electrical specification

### 6.15.4.1 RNG Electrical Specification

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{RNG,START}$ | Time from setting the START task to generation begins. This is a one-time delay on START signal and does not apply between samples. | | 128 | | µs |
| $t_{RNG,RAW}$ | Run time per byte without bias correction. Uniform distribution of 0 and 1 is not guaranteed. | | 30 | | µs |
| $t_{RNG,BC}$ | Run time per byte with bias correction. Uniform distribution of 0 and 1 is guaranteed. Time to generate a byte cannot be guaranteed. | | 120 | | µs |

# 6.16 RTC — Real-time counter

The Real-time counter (RTC) module provides a generic, low power timer on the low-frequency clock source (LFCLK).

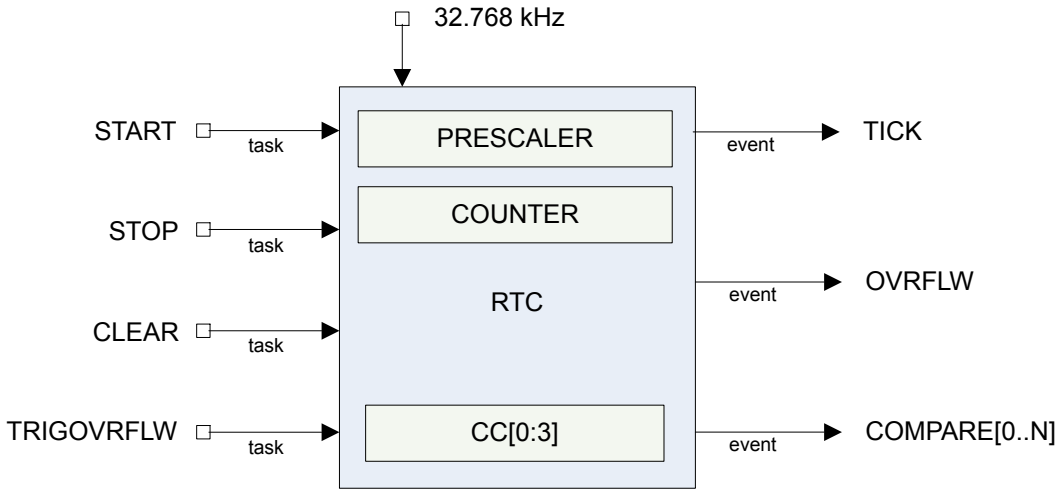



*Figure 89: RTC block schematic*

The RTC module features a 24-bit COUNTER, a 12-bit (1/X) prescaler, capture/compare registers, and a tick event generator for low power, tickless RTOS implementation.

## 6.16.1 Clock source

The RTC will run off the LFCLK.

The COUNTER resolution will therefore be 30.517 µs. Depending on the source, the RTC is able to run while the HFCLK is OFF and PCLK16M is not available.

The software has to explicitely start LFCLK before using the RTC.

See CLOCK — Clock control on page 61 for more information about clock sources.

## 6.16.2 Resolution versus overflow and the PRESCALER

Counter increment frequency:

$$f_{RTC}\ [kHz] = 32.768\ /\ (PRESCALER + 1\ )$$

The PRESCALER register is read/write when the RTC is stopped. The PRESCALER register is read-only once the RTC is STARTed. Writing to the PRESCALER register when the RTC is started has no effect.

The PRESCALER is restarted on START, CLEAR and TRIGOVRFLW, that is, the prescaler value is latched to an internal register (<<PRESC>>) on these tasks.

Examples:

1. Desired COUNTER frequency 100 Hz (10 ms counter period)

   PRESCALER = round(32.768 kHz / 100 Hz) - 1 = 327

   $f_{RTC}$ = 99.9 Hz

   10009.576 µs counter period
2. Desired COUNTER frequency 8 Hz (125 ms counter period)

   PRESCALER = round(32.768 kHz / 8 Hz) – 1 = 4095

   $f_{RTC}$ = 8 Hz

125 ms counter period

| Prescaler | Counter resolution | Overflow |
|---|---|---|
| 0 | 30.517 µs | 512 seconds |
| $2^8$-1 | 7812.5 µs | 131072 seconds |
| $2^{12}$-1 | 125 ms | 582.542 hours |

*Table 72: RTC resolution versus overflow*

## 6.16.3 COUNTER register

The COUNTER increments on LFCLK when the internal PRESCALER register (<<PRESC>>) is 0x00. <<PRESC>> is reloaded from the PRESCALER register. If enabled, the TICK event occurs on each increment of the COUNTER. The TICK event is disabled by default.

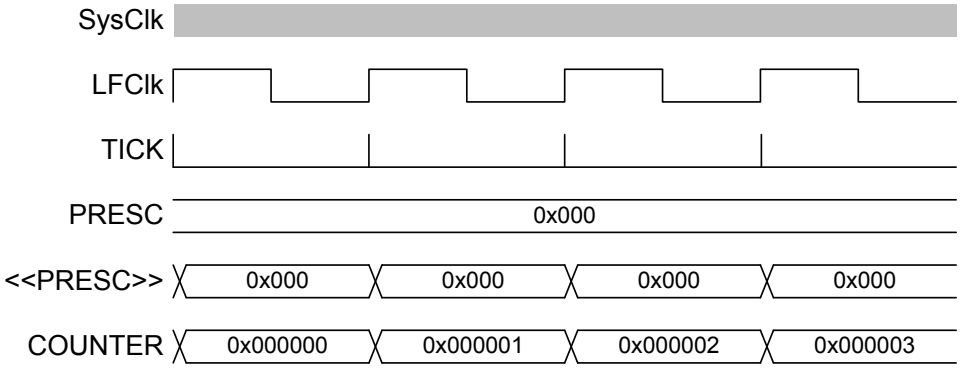



*Figure 90: Timing diagram - COUNTER_PRESCALER_0*

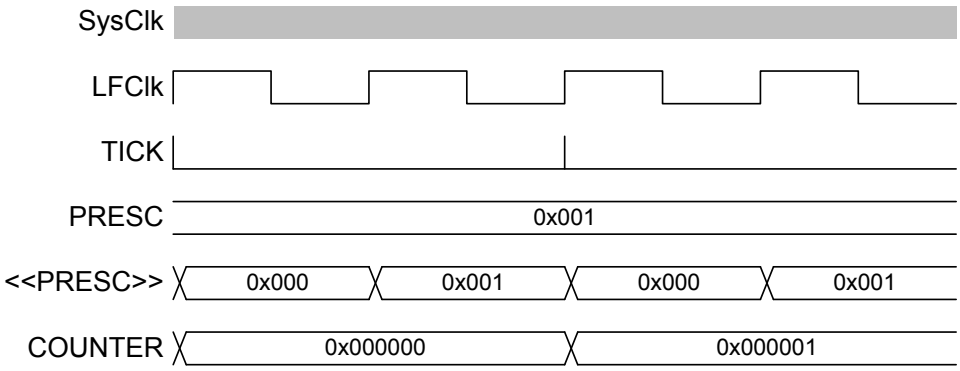



*Figure 91: Timing diagram - COUNTER_PRESCALER_1*

## 6.16.4 Overflow features

The TRIGOVRFLW task sets the COUNTER value to 0xFFFFF0 to allow SW test of the overflow condition.

OVRFLW occurs when COUNTER overflows from 0xFFFFFF to 0.

**Important:** The OVRFLW event is disabled by default.

## 6.16.5 TICK event

The TICK event enables low power "tick-less" RTOS implementation as it optionally provides a regular interrupt source for a RTOS without the need to use the ARM® SysTick feature.

Using the RTC TICK event rather than the SysTick allows the CPU to be powered down while still keeping RTOS scheduling active.

**Important:** The TICK event is disabled by default.

## 6.16.6 Event control feature

To optimize RTC power consumption, events in the RTC can be individually disabled to prevent PCLK16M and HFCLK being requested when those events are triggered. This is managed using the EVTEN register.

For example, if the TICK event is not required for an application, this event should be disabled as it is frequently occurring and may increase power consumption if HFCLK otherwise could be powered down for long durations.

This means that the RTC implements a slightly different task and event system compared to the standard system described in Peripheral interface on page 74. The RTC task and event system is illustrated in Tasks, events and interrupts in the RTC on page 249.

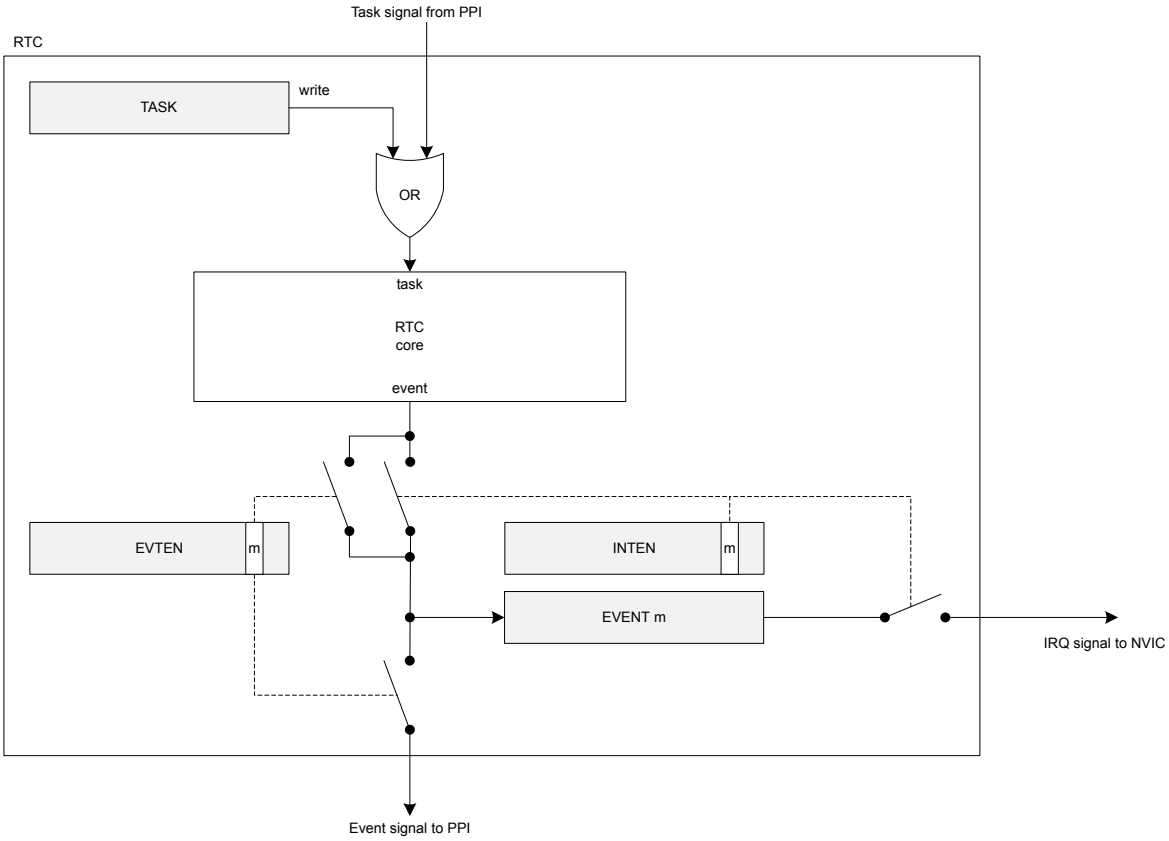



*Figure 92: Tasks, events and interrupts in the RTC*

## 6.16.7 Compare feature

There are a number of Compare registers.

For more information, see Registers on page 254.

When setting a compare register, the following behavior of the RTC compare event should be noted:

- If a CC register value is 0 when a CLEAR task is set, this will not trigger a COMPARE event.

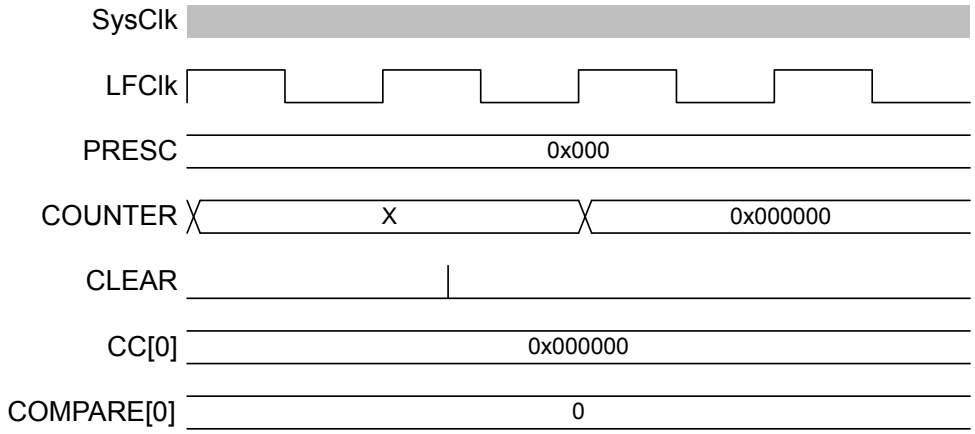



*Figure 93: Timing diagram - COMPARE_CLEAR*

- If a CC register is N and the COUNTER value is N when the START task is set, this will not trigger a COMPARE event.

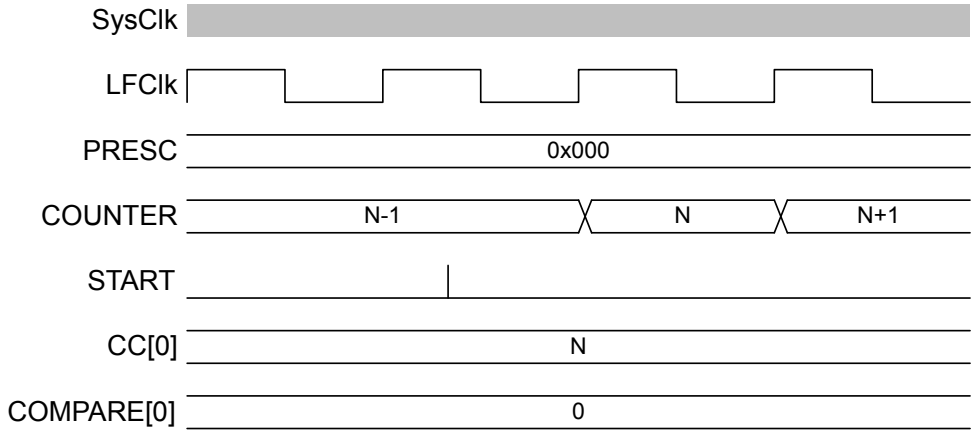



*Figure 94: Timing diagram - COMPARE_START*

- COMPARE occurs when a CC register is N and the COUNTER value transitions from N-1 to N.

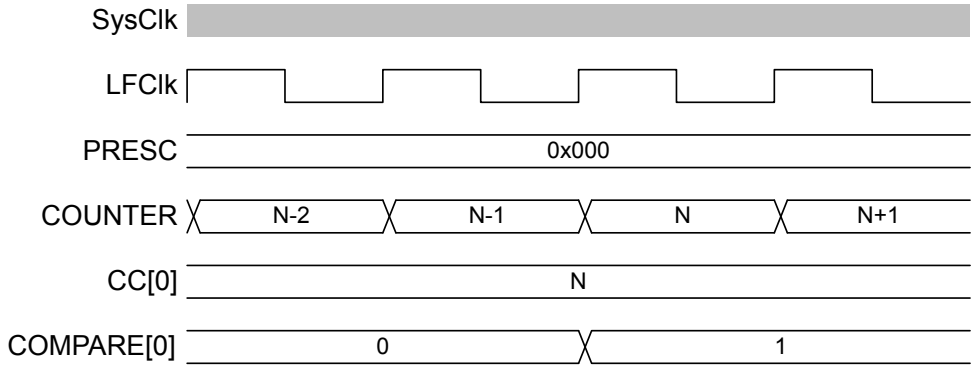



*Figure 95: Timing diagram - COMPARE*

- If the COUNTER is N, writing N+2 to a CC register is guaranteed to trigger a COMPARE event at N+2.

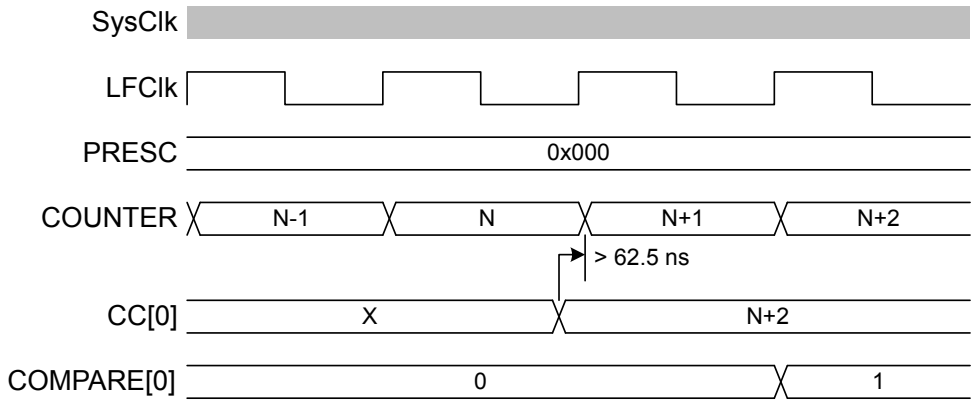



*Figure 96: Timing diagram - COMPARE_N+2*

- If the COUNTER is N, writing N or N+1 to a CC register may not trigger a COMPARE event.

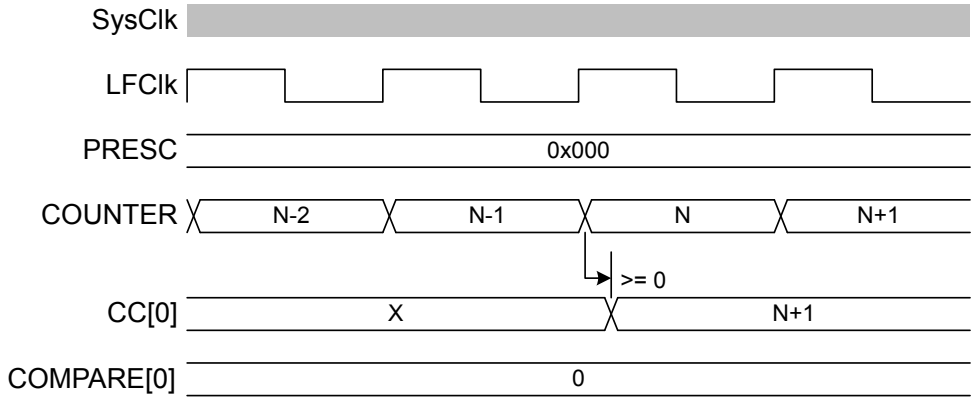



*Figure 97: Timing diagram - COMPARE_N+1*

- If the COUNTER is N and the current CC register value is N+1 or N+2 when a new CC value is written, a match may trigger on the previous CC value before the new value takes effect. If the current CC value greater than N+2 when the new value is written, there will be no event due to the old value.

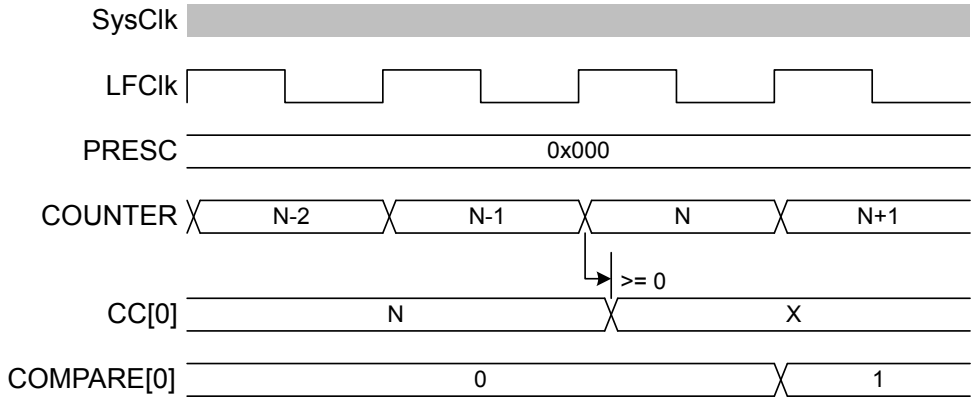



*Figure 98: Timing diagram - COMPARE_N-1*

## 6.16.8 TASK and EVENT jitter/delay

Jitter or delay in the RTC is due to the peripheral clock being a low frequency clock (LFCLK) which is not synchronous to the faster PCLK16M.

Registers in the peripheral interface, part of the PCLK16M domain, have a set of mirrored registers in the LFCLK domain. For example, the COUNTER value accessible from the CPU is in the PCLK16M domain and is latched on read from an internal register called COUNTER in the LFCLK domain. COUNTER is the register which is actually modified each time the RTC ticks. These registers must be synchronised between clock domains (PCLK16M and LFCLK).

The following is a summary of the jitter introduced on tasks and events. Figures illustrating jitter follow.

| Task | Delay |
|---|---|
| CLEAR, STOP, START, TRIGOVRFLOW | +15 to 46 µs |

*Table 73: RTC jitter magnitudes on tasks*

| Operation/Function | Jitter |
|---|---|
| START to COUNTER increment | +/- 15 µs |
| COMPARE to COMPARE [20] | +/- 62.5 ns |

*Table 74: RTC jitter magnitudes on events*

1. CLEAR and STOP (and TRIGOVRFLW; not shown) will be delayed as long as it takes for the peripheral to clock a falling edge and rising of the LFCLK. This is between 15.2585 µs and 45.7755 µs – rounded to 15 µs and 46 µs for the remainder of the section.

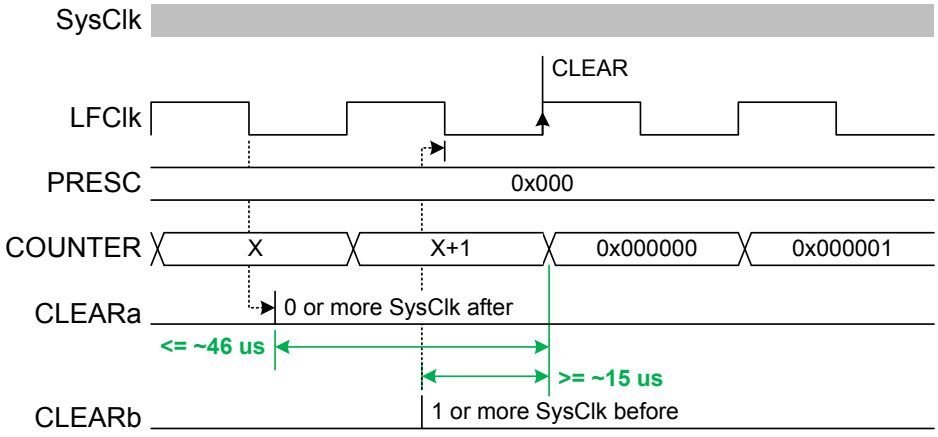



*Figure 99: Timing diagram - DELAY_CLEAR*

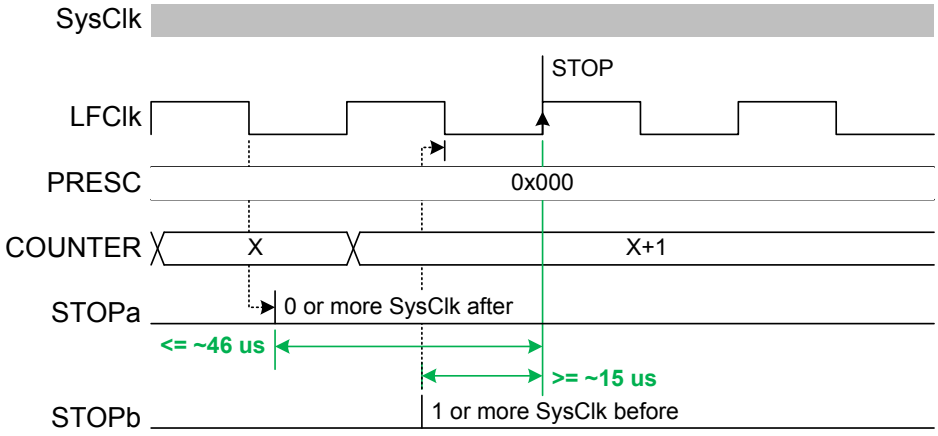



*Figure 100: Timing diagram - DELAY_STOP*

2. The START task will start the RTC. Assuming that the LFCLK was previously running and stable, the first increment of COUNTER (and instance of TICK event) will be typically after 30.5 µs +/-15 µs. In some cases, in particular if the RTC is STARTed before the LFCLK is running, that timing can be up to ~250 µs. The software should therefore wait for the first TICK if it has to make sure the RTC is running.

[20] Assumes RTC runs continuously between these events.

**Note:** 32.768 kHz clock jitter is additional to the numbers provided above.

Sending a TRIGOVRFLW task sets the COUNTER to a value close to overflow. However, since the update of COUNTER relies on a stable LFCLK, sending this task while LFCLK is not running will start LFCLK, but the update will then be delayed by the same amount of time of up to ~250 us. The figures show the smallest and largest delays to on the START task which appears as a +/-15 µs jitter on the first COUNTER increment.

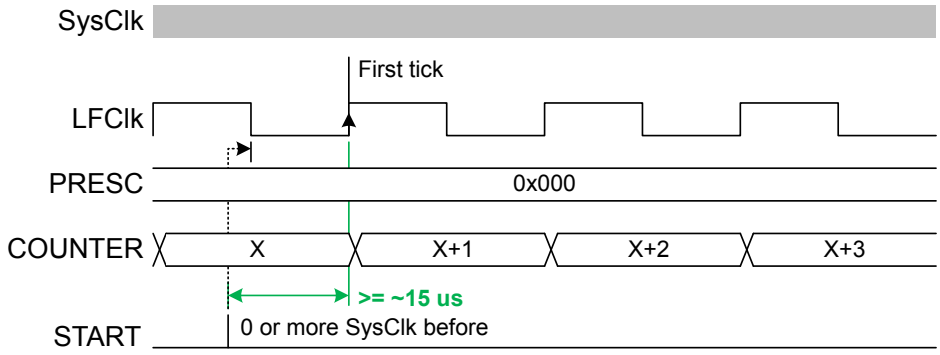



*Figure 101: Timing diagram - JITTER_START-*

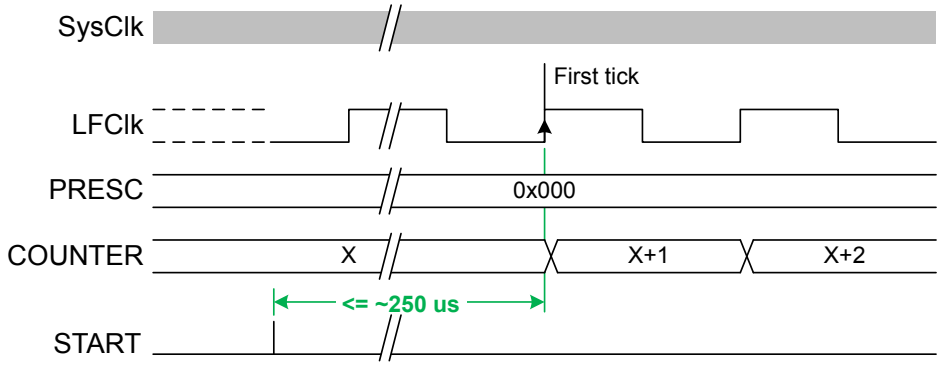



*Figure 102: Timing diagram - JITTER_START+*

## 6.16.9 Reading the COUNTER register

To read the COUNTER register, the internal <<COUNTER>> value is sampled.

To ensure that the <<COUNTER>> is safely sampled (considering an LFCLK transition may occur during a read), the CPU and core memory bus are halted for three cycles by lowering the core PREADY signal. The Read takes the CPU 2 cycles in addition resulting in the COUNTER register read taking a fixed five PCLK16M clock cycles.

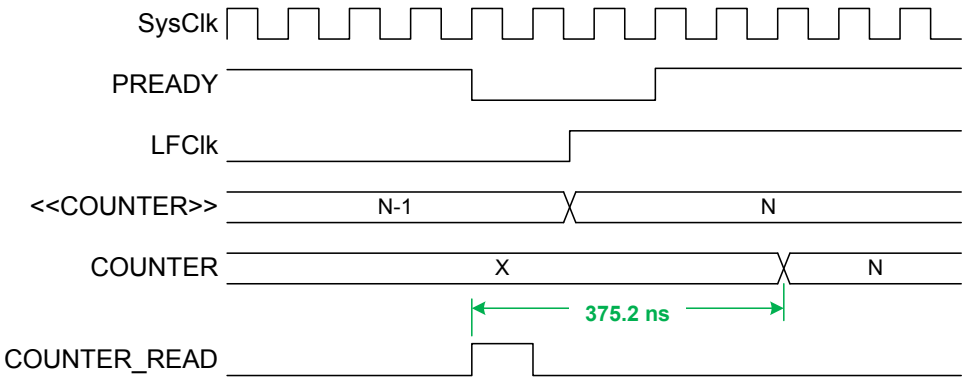



*Figure 103: Timing diagram - COUNTER_READ*

## 6.16.10 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x4000B000 | RTC | RTC0 | Real-time counter 0 | CC[0..2] implemented, CC[3] not implemented |
| 0x40011000 | RTC | RTC1 | Real-time counter 1 | CC[0..3] implemented |

*Table 75: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_START | 0x000 | Start RTC COUNTER |
| TASKS_STOP | 0x004 | Stop RTC COUNTER |
| TASKS_CLEAR | 0x008 | Clear RTC COUNTER |
| TASKS_TRIGOVRFLW | 0x00C | Set COUNTER to 0xFFFFF0 |
| EVENTS_TICK | 0x100 | Event on COUNTER increment |
| EVENTS_OVRFLW | 0x104 | Event on COUNTER overflow |
| EVENTS_COMPARE[0] | 0x140 | Compare event on CC[0] match |
| EVENTS_COMPARE[1] | 0x144 | Compare event on CC[1] match |
| EVENTS_COMPARE[2] | 0x148 | Compare event on CC[2] match |
| EVENTS_COMPARE[3] | 0x14C | Compare event on CC[3] match |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| EVTEN | 0x340 | Enable or disable event routing |
| EVTENSET | 0x344 | Enable event routing |
| EVTENCLR | 0x348 | Disable event routing |
| COUNTER | 0x504 | Current COUNTER value |
| PRESCALER | 0x508 | 12 bit prescaler for COUNTER frequency (32768/(PRESCALER+1)).Must be written when RTC is stopped |
| CC[0] | 0x540 | Compare register 0 |
| CC[1] | 0x544 | Compare register 1 |
| CC[2] | 0x548 | Compare register 2 |
| CC[3] | 0x54C | Compare register 3 |

*Table 76: Register overview*

### 6.16.10.1 TASKS_START

Address offset: 0x000

Start RTC COUNTER

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_START | | | Start RTC COUNTER |
| | | | Trigger | 1 | Trigger task |

### 6.16.10.2 TASKS_STOP

Address offset: 0x004

Stop RTC COUNTER

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STOP | | | Stop RTC COUNTER |
| | | | Trigger | 1 | Trigger task |

### 6.16.10.3 TASKS_CLEAR

Address offset: 0x008

Clear RTC COUNTER

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_CLEAR | | | Clear RTC COUNTER |
| | | | Trigger | 1 | Trigger task |

### 6.16.10.4 TASKS_TRIGOVRFLW

Address offset: 0x00C

Set COUNTER to 0xFFFFF0

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_TRIGOVRFLW | | | Set COUNTER to 0xFFFFF0 |
| | | | Trigger | 1 | Trigger task |

### 6.16.10.5 EVENTS_TICK

Address offset: 0x100

Event on COUNTER increment

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_TICK | | | Event on COUNTER increment |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.16.10.6 EVENTS_OVRFLW

Address offset: 0x104

Event on COUNTER overflow

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_OVRFLW | | | Event on COUNTER overflow |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.16.10.7 EVENTS_COMPARE[n] (n=0..3)

Address offset: 0x140 + (n × 0x4)

Compare event on CC[n] match

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_COMPARE | | | Compare event on CC[n] match |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.16.10.8 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | TICK | | | Write '1' to enable interrupt for event TICK |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | OVRFLW | | | Write '1' to enable interrupt for event OVRFLW |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C-F | RW | COMPARE[i] (i=0..3) | | | Write '1' to enable interrupt for event COMPARE[i] |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.16.10.9 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | TICK | | | Write '1' to disable interrupt for event TICK |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | OVRFLW | | | Write '1' to disable interrupt for event OVRFLW |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C-F | RW | COMPARE[i] (i=0..3) | | | Write '1' to disable interrupt for event COMPARE[i] |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.16.10.10 EVTEN

Address offset: 0x340

Enable or disable event routing

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | TICK | | | Enable or disable event routing for event TICK |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Disable |
| B | RW | OVRFLW | | | Enable or disable event routing for event OVRFLW |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Disable |
| C-F | RW | COMPARE[i] (i=0..3) | | | Enable or disable event routing for event COMPARE[i] |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Disable |

## 6.16.10.11 EVTENSET

Address offset: 0x344

Enable event routing

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | TICK | | | Write '1' to enable event routing for event TICK |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| | | | Set | 1 | Enable |
| B | RW | OVRFLW | | | Write '1' to enable event routing for event OVRFLW |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| | | | Set | 1 | Enable |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| C-F | RW | COMPARE[i] (i=0..3) | | | Write '1' to enable event routing for event COMPARE[i] |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| | | | Set | 1 | Enable |

## 6.16.10.12 EVTENCLR

Address offset: 0x348

Disable event routing

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | TICK | | | Write '1' to disable event routing for event TICK |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| | | | Clear | 1 | Disable |
| B | RW | OVRFLW | | | Write '1' to disable event routing for event OVRFLW |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| | | | Clear | 1 | Disable |
| C-F | RW | COMPARE[i] (i=0..3) | | | Write '1' to disable event routing for event COMPARE[i] |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| | | | Clear | 1 | Disable |

## 6.16.10.13 COUNTER

Address offset: 0x504

Current COUNTER value

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | COUNTER | | | Counter value |

## 6.16.10.14 PRESCALER

Address offset: 0x508

12 bit prescaler for COUNTER frequency (32768/(PRESCALER+1)).Must be written when RTC is stopped

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PRESCALER | | | Prescaler value |

### 6.16.10.15 CC[n] (n=0..3)

Address offset: 0x540 + (n × 0x4)

Compare register n

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | COMPARE | | | Compare value |

## 6.16.11 Electrical specification

# 6.17 SAADC — Successive approximation analog-to-digital converter

The ADC is a differential successive approximation register (SAR) analog-to-digital converter.

Listed here are the main features of SAADC:

- 8/10/12-bit resolution, 14-bit resolution with oversampling
- Up to eight input channels
  - One channel per single-ended input and two channels per differential input
  - Scan mode can be configured with both single-ended channels and differential channels.
- Full scale input range (0 to VDD)
- Sampling triggered via a task from software or a PPI channel for full flexibility on sample frequency source from low power 32.768kHz RTC or more accurate 1/16MHz Timers
- One-shot conversion mode to sample a single channel
- Scan mode to sample a series of channels in sequence. Sample delay between channels is $t_{ack}$ + $t_{conv}$ which may vary between channels according to user configuration of $t_{ack}$.
- Support for direct sample transfer to RAM using EasyDMA
- Interrupts on single sample and full buffer events
- Samples stored as 16-bit 2's complement values for differential and single-ended sampling
- Continuous sampling without the need of an external timer
- Internal resistor string
- Limit checking on the fly

## 6.17.1 Shared resources

The ADC can coexist with COMP and other peripherals using one of `AIN0–AIN7`, provided these are assigned to different pins.

It is not recommended to select the same analog input pin for both modules.

## 6.17.2 Overview

The ADC supports up to eight external analog input channels, depending on package variant. It can be operated in a one-shot mode with sampling under software control, or a continuous conversion mode with a programmable sampling rate.

The analog inputs can be configured as eight single-ended inputs, four differential inputs or a combination of these. Each channel can be configured to select `AIN0` to `AIN7` pins, or the `VDD` pin. Channels can be

sampled individually in one-shot or continuous sampling modes, or, using scan mode, multiple channels can be sampled in sequence. Channels can also be oversampled to improve noise performance.

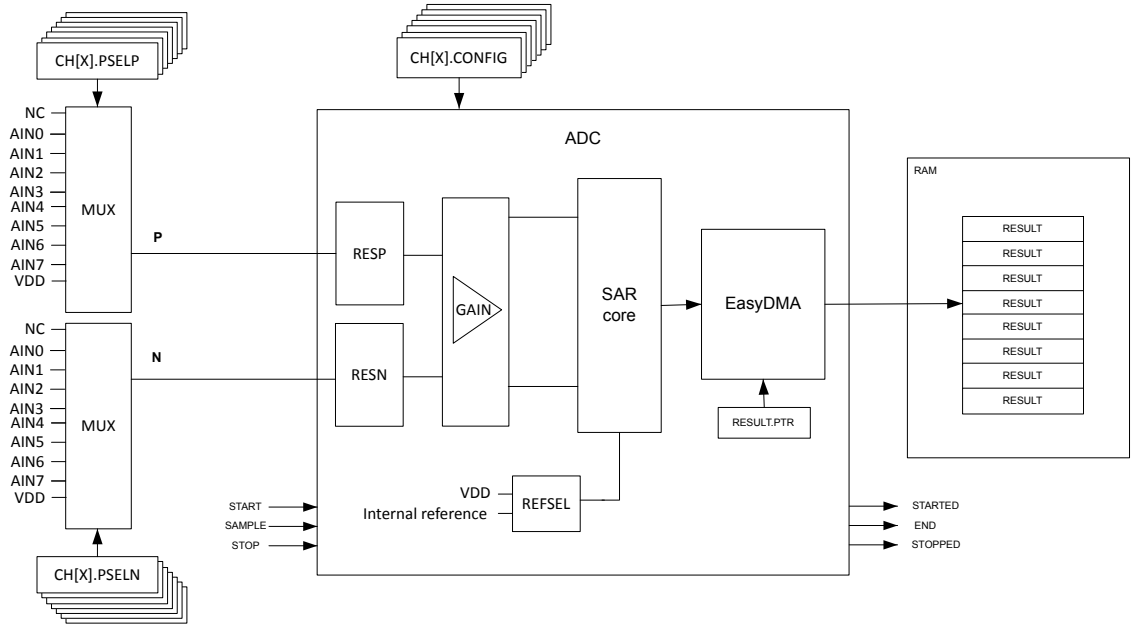



*Figure 104: Simplified ADC block diagram*

Internally, the ADC is always a differential analog-to-digital converter, but by default it is configured with single-ended input in the MODE field of the CH[n].CONFIG register. In single-ended mode, the negative input will be shorted to ground internally.

The assumption in single-ended mode is that the internal ground of the ADC is the same as the external ground that the measured voltage is referred to. The ADC is thus sensitive to ground bounce on the PCB in single-ended mode. If this is a concern we recommend using differential measurement.

## 6.17.3 Digital output

The output result of the ADC depends on the settings in the CH[n].CONFIG and RESOLUTION registers as follows:

$$RESULT = [V(P) - V(N)] * GAIN/REFERENCE * 2^{(RESOLUTION - m)}$$

where

**V(P)**

is the voltage at input P

**V(N)**

is the voltage at input N

**GAIN**

is the selected gain setting

**REFERENCE**

is the selected reference voltage

and m=0 if CONFIG.MODE=SE, or m=1 if CONFIG.MODE=Diff.

The result generated by the ADC will deviate from the expected due DC errors like offset, gain, differential non-linearity (DNL), and integral non-linearity (INL). See Electrical specification for details on these parameters. The result can also vary due to AC errors like non-linearities in the GAIN block, settling errors

due to high source impedance and sampling jitter. For battery measurement the DC errors are most noticeable.

The ADC has a wide selection of gains controlled in the GAIN field of the CH[n].CONFIG register. If CH[n].CONFIG.REFSEL=0, the input range of the ADC core is nominally ±0.6 V differential and the input must be scaled accordingly.

The ADC has a temperature dependent offset. If the ADC is to operate over a large temperature range, we recommend running CALIBRATEOFFSET at regular intervals, a CALIBRATEDONE event will be fired when the calibration is complete

## 6.17.4 Analog inputs and channels

Up to eight analog input channels, CH[n](n=0..7), can be configured.

See Shared resources on page 259 for shared input with comparators.

Any one of the available channels can be enabled for the ADC to operate in one-shot mode. If more than one CH[n] is configured, the ADC enters scan mode.

An analog input is selected as a positive converter input if CH[n].PSELP is set, setting CH[n].PSELP also enables the particular channel.

An analog input is selected as a negative converter input if CH[n].PSELN is set. The CH[n].PSELN register will have no effect unless differential mode is enabled, see MODE field in CH[n].CONFIG register.

If more than one of the CH[n].PSELP registers is set, the device enters scan mode. Input selections in scan mode are controlled by the CH[n].PSELP and CH[n].PSELN registers, where CH[n].PSELN is only used if the particular scan channel is specified as differential, see MODE field in CH[n].CONFIG register.

**Important:** Channels selected for COMP cannot be used at the same time for ADC sampling, though channels not selected for use by these blocks can be used by the ADC.

| Channel input | Source | Connectivity |
|---|---|---|
| CH[n].PSELP | AIN0...AIN7 | Yes(any) |
| CH[n].PSELP | VDD | Yes |
| CH[n].PSELN | AIN0...AIN7 | Yes(any) |
| CH[n].PSELN | VDD | Yes |

*Table 77: Legal connectivity CH[n] vs. analog input*

## 6.17.5 Operation modes

The ADC input configuration supports one-shot mode, continuous mode and scan mode.

Scan mode and oversampling cannot be combined.

### 6.17.5.1 One-shot mode

One-shot operation is configured by enabling only one of the available channels defined by CH[n].PSELP, CH[n].PSELN, and CH[n].CONFIG registers.

Upon a SAMPLE task, the ADC starts to sample the input voltage. The CH[n].CONFIG.TACQ controls the acquisition time.

A DONE event signals that one sample has been taken.

In this mode, the RESULTDONE event has the same meaning as DONE when no oversampling takes place. Note that both events may occur before the actual value has been transferred into RAM by EasyDMA. For more information, see EasyDMA on page 263.

### 6.17.5.2 Continuous mode

Continuous sampling can be achieved by using the internal timer in the ADC, or triggering the SAMPLE task from one of the general purpose timers through the PPI.

Care shall be taken to ensure that the sample rate fulfils the following criteria, depending on how many channels are active:

$f_{SAMPLE} < 1/[t_{ACQ} + t_{conv}]$

The SAMPLERATE register can be used as a local timer instead of triggering individual SAMPLE tasks. When SAMPLERATE.MODE is set to Timers, it is sufficient to trigger SAMPLE task only once in order to start the SAADC and triggering the STOP task will stop sampling. The SAMPLERATE.CC field controls the sample rate.

The SAMPLERATE timer mode cannot be combined with SCAN mode, and only one channel can be enabled in this mode.

A DONE event signals that one sample has been taken.

In this mode, the RESULTDONE event has the same meaning as DONE when no oversampling takes place. Note that both events may occur before the actual value has been transferred into RAM by EasyDMA.

### 6.17.5.3 Oversampling

An accumulator in the ADC can be used to average noise on the analog input. In general, oversampling improves the signal-to-noise ratio (SNR). Oversampling, however, does not improve the integral non-linearity (INL), or differential non-linearity (DNL).

Oversampling and scan should not be combined, since oversampling and scan will average over input channels.

The accumulator is controlled in the OVERSAMPLE register. The SAMPLE task must be set $2^{OVERSAMPLE}$ number of times before the result is written to RAM. This can be achieved by:

- Configuring a fixed sampling rate using the local timer or a general purpose timer and PPI to trigger a SAMPLE task
- Triggering SAMPLE $2^{OVERSAMPLE}$ times from software
- Enabling BURST mode

CH[n].CONFIG.BURST can be enabled to avoid setting SAMPLE task $2^{OVERSAMPLE}$ times. With BURST = 1 the ADC will sample the input $2^{OVERSAMPLE}$ times as fast as it can (actual timing: $<(t_{ACQ}+t_{CONV})\times 2^{OVERSAMPLE}$). Thus, for the user it will just appear like the conversion took a bit longer time, but other than that, it is similar to one-shot mode. Scan mode can be combined with BURST=1, if burst is enabled on all channels.

A DONE event signals that one sample has been taken.

In this mode, the RESULTDONE event signals that enough conversions have taken place for an oversampled result to get transferred into RAM. Note that both events may occur before the actual value has been transferred into RAM by EasyDMA.

### 6.17.5.4 Scan mode

A channel is considered enabled if CH[n].PSELP is set. If more than one channel, CH[n], is enabled, the ADC enters scan mode.

In scan mode, one SAMPLE task will trigger one conversion per enabled channel. The time it takes to sample all channels is:

$\text{Total time} < \text{Sum}(CH[x].t_{ACQ}+t_{CONV}),\ x=0..\text{enabled channels}$

A DONE event signals that one sample has been taken.

In this mode, the RESULTDONE event signals has the same meaning as DONE when no oversampling takes place. Note that both events may occur before the actual values have been transferred into RAM by EasyDMA.

Example of RAM placement (even RESULT.MAXCNT), channels 1, 2 and 5 enabled on page 263 provides an example of results placement in Data RAM, with an even RESULT.MAXCNT. In this example, channels 1, 2 and 5 are enabled, all others are disabled.

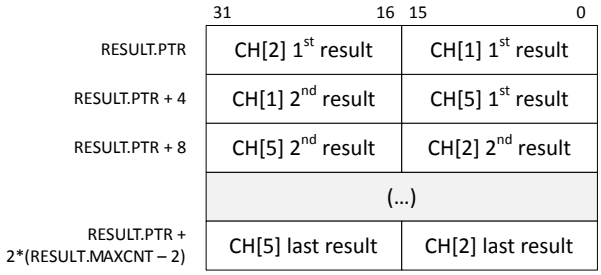



*Figure 105: Example of RAM placement (even RESULT.MAXCNT), channels 1, 2 and 5 enabled*

Example of RAM placement (odd RESULT.MAXCNT), channels 1, 2 and 5 enabled on page 263 provides an example of results placement in Data RAM, with an odd RESULT.MAXCNT. In this example, channels 1, 2 and 5 are enabled, all others are disabled. The last 32-bit word is populated only with one 16-bit result.

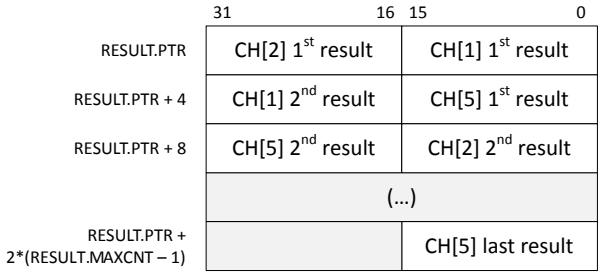



*Figure 106: Example of RAM placement (odd RESULT.MAXCNT), channels 1, 2 and 5 enabled*

## 6.17.6 EasyDMA

After configuring RESULT.PTR and RESULT.MAXCNT, the ADC resources are started by triggering the START task. The ADC is using EasyDMA to store results in a Result buffer in RAM.

The Result buffer is located at the address specified in the RESULT.PTR register. The RESULT.PTR register is double-buffered and it can be updated and prepared for the next START task immediately after the STARTED event is generated. The size of the Result buffer is specified in the RESULT.MAXCNT register and the ADC will generate an END event when it has filled up the Result buffer, see ADC on page 264. Results are stored in little-endian byte order in Data RAM. Every sample will be sign extended to 16 bit before stored in the Result buffer.

The ADC is stopped by triggering the STOP task. The STOP task will terminate an ongoing sampling. The ADC will generate a STOPPED event when it has stopped. If the ADC is already stopped when the STOP task is triggered, the STOPPED event will still be generated.

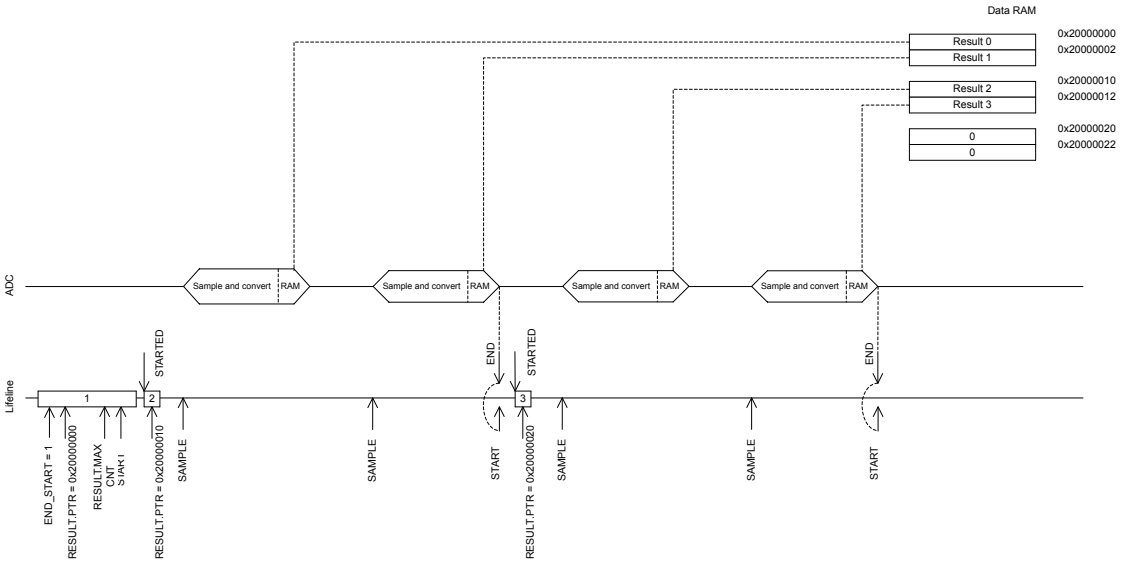



*Figure 107: ADC*

If the RESULT.PTR is not pointing to the Data RAM region, an EasyDMA transfer may result in a HardFault or RAM corruption. See Memory on page 16 for more information about the different memory regions.

The EasyDMA will have finished accessing the RAM when the END or STOPPED event has been generated.

The RESULT.AMOUNT register can be read following an END event or a STOPPED event to see how many results have been transferred to the Result buffer in RAM since the START task was triggered.

In scan mode, SAMPLE tasks can be triggered once the START task is triggered. The END event is generated when the number of samples transferred to memory reaches the value specified by RESULT.MAXCNT. After an END event, the START task needs to be triggered again before new samples can be taken. Also make sure that the size of the Result buffer is large enough to have space for minimum one result from each of the enabled channels, by specifying RESULT.MAXCNT >= number of channels enabled. For more information about the scan mode, see Scan mode on page 262.

## 6.17.7 Resistor ladder

The ADC has an internal resistor string for positive and negative input.

See Resistor ladder for positive input (negative input is equivalent, using RESN instead of RESP) on page 265. The resistors are controlled in the CH[n].CONFIG.RESP and CH[n].CONFIG.RESN registers.

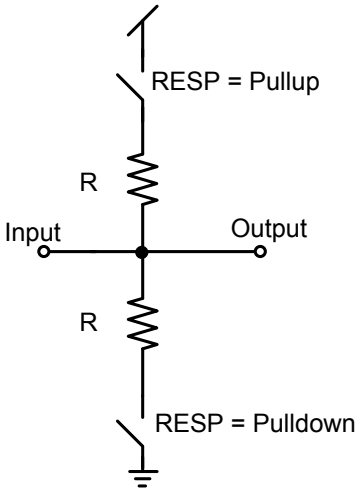



*Figure 108: Resistor ladder for positive input (negative input is equivalent, using RESN instead of RESP)*

## 6.17.8 Reference

The ADC can use two different references, controlled in the REFSEL field of the CH[n].CONFIG register.

These are:

- Internal reference
- VDD as reference

The internal reference results in an input range of ±0.6 V on the ADC core. VDD as reference results in an input range of ±VDD/4 on the ADC core. The gain block can be used to change the effective input range of the ADC.

```
Input range = (+- 0.6 V or +-VDD/4)/Gain
```

For example, choosing VDD as reference, single ended input (grounded negative input), and a gain of 1/4 the input range will be:

```
Input range = (VDD/4)/(1/4) = VDD
```

With internal reference, single ended input (grounded negative input), and a gain of 1/6 the input range will be:

```
Input range = (0.6 V)/(1/6) = 3.6 V
```

The AIN0-AIN7 inputs cannot exceed VDD, or be lower than VSS.

## 6.17.9 Acquisition time

To sample the input voltage, the ADC connects a capacitor to the input.

For illustration, see Simplified ADC sample network on page 266. The acquisition time indicates how long the capacitor is connected, see TACQ field in CH[n].CONFIG register. The required acquisition time depends on the source ($R_{source}$) resistance. For high source resistance the acquisition time should be increased, see Acquisition time on page 266.

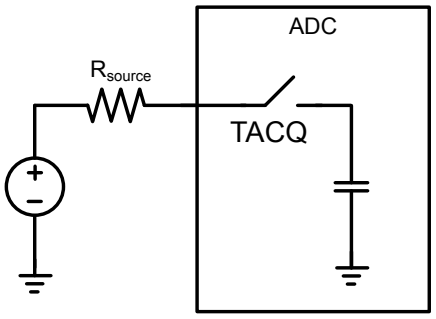



*Figure 109: Simplified ADC sample network*

| TACQ [µs] | Maximum source resistance [kOhm] |
|---|---|
| 3 | 10 |
| 5 | 40 |
| 10 | 100 |
| 15 | 200 |
| 20 | 400 |
| 40 | 800 |

*Table 78: Acquisition time*

## 6.17.10 Limits event monitoring

A channel can be event monitored by configuring limit register CH[n].LIMIT.

If the conversion result is higher than the defined high limit, or lower than the defined low limit, the appropriate event will get fired.

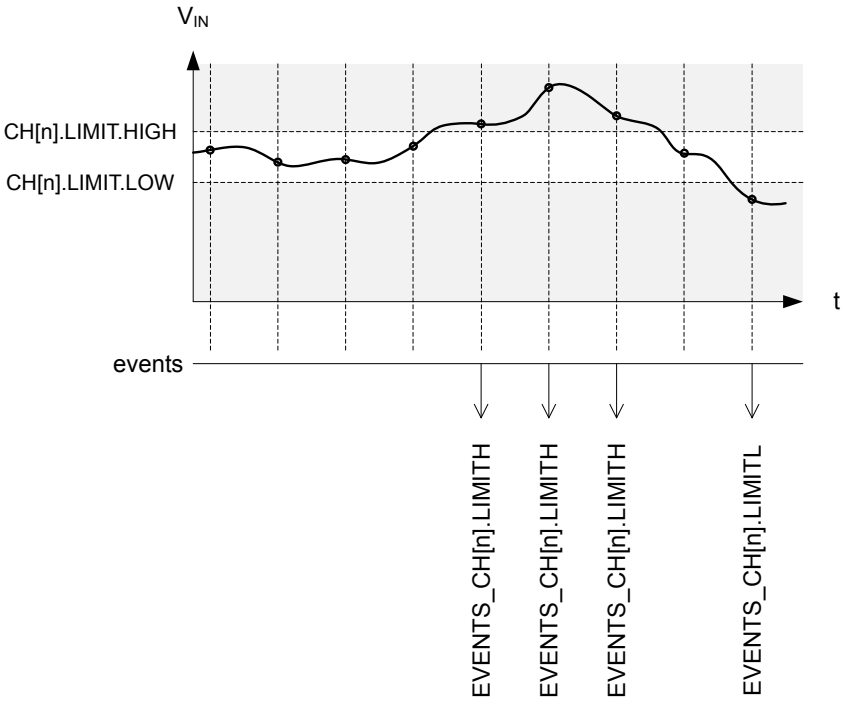



*Figure 110: Example of limits monitoring on channel 'n'*

Note that when setting the limits, CH[n].LIMIT.HIGH shall always be higher than or equal to CH[n].LIMIT.LOW . In other words, an event can be fired only when the input signal has been sampled outside of the defined limits. It is not possible to fire an event when the input signal is inside a defined range by swapping high and low limits.

The comparison to limits always takes place, there is no need to enable it. If comparison is not required on a channel, the software shall simply ignore the related events. In that situation, the value of the limits registers is irrelevant, so it does not matter if CH[n].LIMIT.LOW is lower than CH[n].LIMIT.HIGH or not.

## 6.17.11 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40007000 | SAADC | SAADC | Analog-to-digital converter | |

*Table 79: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_START | 0x000 | Start the ADC and prepare the result buffer in RAM |
| TASKS_SAMPLE | 0x004 | Take one ADC sample, if scan is enabled all channels are sampled |
| TASKS_STOP | 0x008 | Stop the ADC and terminate any on-going conversion |
| TASKS_CALIBRATEOFFSET | 0x00C | Starts offset auto-calibration |
| EVENTS_STARTED | 0x100 | The ADC has started |
| EVENTS_END | 0x104 | The ADC has filled up the Result buffer |
| EVENTS_DONE | 0x108 | A conversion task has been completed. Depending on the mode, multiple conversions might be needed for a result to be transferred to RAM. |
| EVENTS_RESULTDONE | 0x10C | A result is ready to get transferred to RAM. |
| EVENTS_CALIBRATEDONE | 0x110 | Calibration is complete |
| EVENTS_STOPPED | 0x114 | The ADC has stopped |
| EVENTS_CH[0].LIMITH | 0x118 | Last results is equal or above CH[0].LIMIT.HIGH |
| EVENTS_CH[0].LIMITL | 0x11C | Last results is equal or below CH[0].LIMIT.LOW |
| EVENTS_CH[1].LIMITH | 0x120 | Last results is equal or above CH[1].LIMIT.HIGH |
| EVENTS_CH[1].LIMITL | 0x124 | Last results is equal or below CH[1].LIMIT.LOW |
| EVENTS_CH[2].LIMITH | 0x128 | Last results is equal or above CH[2].LIMIT.HIGH |
| EVENTS_CH[2].LIMITL | 0x12C | Last results is equal or below CH[2].LIMIT.LOW |
| EVENTS_CH[3].LIMITH | 0x130 | Last results is equal or above CH[3].LIMIT.HIGH |
| EVENTS_CH[3].LIMITL | 0x134 | Last results is equal or below CH[3].LIMIT.LOW |
| EVENTS_CH[4].LIMITH | 0x138 | Last results is equal or above CH[4].LIMIT.HIGH |
| EVENTS_CH[4].LIMITL | 0x13C | Last results is equal or below CH[4].LIMIT.LOW |
| EVENTS_CH[5].LIMITH | 0x140 | Last results is equal or above CH[5].LIMIT.HIGH |
| EVENTS_CH[5].LIMITL | 0x144 | Last results is equal or below CH[5].LIMIT.LOW |
| EVENTS_CH[6].LIMITH | 0x148 | Last results is equal or above CH[6].LIMIT.HIGH |
| EVENTS_CH[6].LIMITL | 0x14C | Last results is equal or below CH[6].LIMIT.LOW |
| EVENTS_CH[7].LIMITH | 0x150 | Last results is equal or above CH[7].LIMIT.HIGH |
| EVENTS_CH[7].LIMITL | 0x154 | Last results is equal or below CH[7].LIMIT.LOW |
| INTEN | 0x300 | Enable or disable interrupt |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| STATUS | 0x400 | Status |
| ENABLE | 0x500 | Enable or disable ADC |
| CH[0].PSELP | 0x510 | Input positive pin selection for CH[0] |
| CH[0].PSELN | 0x514 | Input negative pin selection for CH[0] |
| CH[0].CONFIG | 0x518 | Input configuration for CH[0] |
| CH[0].LIMIT | 0x51C | High/low limits for event monitoring a channel |
| CH[1].PSELP | 0x520 | Input positive pin selection for CH[1] |

| Register | Offset | Description |
|---|---|---|
| CH[1].PSELN | 0x524 | Input negative pin selection for CH[1] |
| CH[1].CONFIG | 0x528 | Input configuration for CH[1] |
| CH[1].LIMIT | 0x52C | High/low limits for event monitoring a channel |
| CH[2].PSELP | 0x530 | Input positive pin selection for CH[2] |
| CH[2].PSELN | 0x534 | Input negative pin selection for CH[2] |
| CH[2].CONFIG | 0x538 | Input configuration for CH[2] |
| CH[2].LIMIT | 0x53C | High/low limits for event monitoring a channel |
| CH[3].PSELP | 0x540 | Input positive pin selection for CH[3] |
| CH[3].PSELN | 0x544 | Input negative pin selection for CH[3] |
| CH[3].CONFIG | 0x548 | Input configuration for CH[3] |
| CH[3].LIMIT | 0x54C | High/low limits for event monitoring a channel |
| CH[4].PSELP | 0x550 | Input positive pin selection for CH[4] |
| CH[4].PSELN | 0x554 | Input negative pin selection for CH[4] |
| CH[4].CONFIG | 0x558 | Input configuration for CH[4] |
| CH[4].LIMIT | 0x55C | High/low limits for event monitoring a channel |
| CH[5].PSELP | 0x560 | Input positive pin selection for CH[5] |
| CH[5].PSELN | 0x564 | Input negative pin selection for CH[5] |
| CH[5].CONFIG | 0x568 | Input configuration for CH[5] |
| CH[5].LIMIT | 0x56C | High/low limits for event monitoring a channel |
| CH[6].PSELP | 0x570 | Input positive pin selection for CH[6] |
| CH[6].PSELN | 0x574 | Input negative pin selection for CH[6] |
| CH[6].CONFIG | 0x578 | Input configuration for CH[6] |
| CH[6].LIMIT | 0x57C | High/low limits for event monitoring a channel |
| CH[7].PSELP | 0x580 | Input positive pin selection for CH[7] |
| CH[7].PSELN | 0x584 | Input negative pin selection for CH[7] |
| CH[7].CONFIG | 0x588 | Input configuration for CH[7] |
| CH[7].LIMIT | 0x58C | High/low limits for event monitoring a channel |
| RESOLUTION | 0x5F0 | Resolution configuration |
| OVERSAMPLE | 0x5F4 | Oversampling configuration. OVERSAMPLE should not be combined with SCAN. The RESOLUTION is applied before averaging, thus for high OVERSAMPLE a higher RESOLUTION should be used. |
| SAMPLERATE | 0x5F8 | Controls normal or continuous sample rate |
| RESULT.PTR | 0x62C | Data pointer |
| RESULT.MAXCNT | 0x630 | Maximum number of buffer words to transfer |
| RESULT.AMOUNT | 0x634 | Number of buffer words transferred since last START |

*Table 80: Register overview*

## 6.17.11.1 TASKS_START

Address offset: 0x000

Start the ADC and prepare the result buffer in RAM

| Bit number | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | A |
| **Reset 0x00000000** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_START | | | Start the ADC and prepare the result buffer in RAM |
| | | | Trigger | 1 | Trigger task |

## 6.17.11.2 TASKS_SAMPLE

Address offset: 0x004

Take one ADC sample, if scan is enabled all channels are sampled

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_SAMPLE | | | Take one ADC sample, if scan is enabled all channels are sampled |
| | | | Trigger | 1 | Trigger task |

## 6.17.11.3 TASKS_STOP

Address offset: 0x008

Stop the ADC and terminate any on-going conversion

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STOP | | | Stop the ADC and terminate any on-going conversion |
| | | | Trigger | 1 | Trigger task |

## 6.17.11.4 TASKS_CALIBRATEOFFSET

Address offset: 0x00C

Starts offset auto-calibration

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_CALIBRATEOFFSET | | | Starts offset auto-calibration |
| | | | Trigger | 1 | Trigger task |

## 6.17.11.5 EVENTS_STARTED

Address offset: 0x100

The ADC has started

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_STARTED | | | The ADC has started |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.17.11.6 EVENTS_END

Address offset: 0x104

The ADC has filled up the Result buffer

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_END | | | The ADC has filled up the Result buffer |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.17.11.7 EVENTS_DONE

Address offset: 0x108

A conversion task has been completed. Depending on the mode, multiple conversions might be needed for a result to be transferred to RAM.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_DONE | | | A conversion task has been completed. Depending on the mode, multiple conversions might be needed for a result to be transferred to RAM. |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.17.11.8 EVENTS_RESULTDONE

Address offset: 0x10C

A result is ready to get transferred to RAM.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_RESULTDONE | | | A result is ready to get transferred to RAM. |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.17.11.9 EVENTS_CALIBRATEDONE

Address offset: 0x110

Calibration is complete

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_CALIBRATEDONE | | | Calibration is complete |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.17.11.10 EVENTS_STOPPED

Address offset: 0x114

The ADC has stopped

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_STOPPED | | | The ADC has stopped |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.17.11.11 EVENTS_CH[n].LIMITH (n=0..7)

Address offset: 0x118 + (n × 0x8)

Last results is equal or above CH[n].LIMIT.HIGH

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | LIMITH | | | Last results is equal or above CH[n].LIMIT.HIGH |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.17.11.12 EVENTS_CH[n].LIMITL (n=0..7)

Address offset: 0x11C + (n × 0x8)

Last results is equal or below CH[n].LIMIT.LOW

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | LIMITL | | | Last results is equal or below CH[n].LIMIT.LOW |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.17.11.13 INTEN

Address offset: 0x300

Enable or disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | STARTED | | | Enable or disable interrupt for event STARTED |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| B | RW | END | | | Enable or disable interrupt for event END |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| C | RW | DONE | | | Enable or disable interrupt for event DONE |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| D | RW | RESULTDONE | | | Enable or disable interrupt for event RESULTDONE |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| E | RW | CALIBRATEDONE | | | Enable or disable interrupt for event CALIBRATEDONE |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| F | RW | STOPPED | | | Enable or disable interrupt for event STOPPED |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| G | RW | CH0LIMITH | | | Enable or disable interrupt for event CH0LIMITH |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| H | RW | CH0LIMITL | | | Enable or disable interrupt for event CH0LIMITL |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| I | RW | CH1LIMITH | | | Enable or disable interrupt for event CH1LIMITH |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| J | RW | CH1LIMITL | | | Enable or disable interrupt for event CH1LIMITL |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| K | RW | CH2LIMITH | | | Enable or disable interrupt for event CH2LIMITH |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| L | RW | CH2LIMITL | | | Enable or disable interrupt for event CH2LIMITL |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| M | RW | CH3LIMITH | | | Enable or disable interrupt for event CH3LIMITH |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| N | RW | CH3LIMITL | | | Enable or disable interrupt for event CH3LIMITL |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| O | RW | CH4LIMITH | | | Enable or disable interrupt for event CH4LIMITH |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| P | RW | CH4LIMITL | | | Enable or disable interrupt for event CH4LIMITL |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| Q | RW | CH5LIMITH | | | Enable or disable interrupt for event CH5LIMITH |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| R | RW | CH5LIMITL | | | Enable or disable interrupt for event CH5LIMITL |
| | | | Disabled | 0 | Disable |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| | | | Enabled | 1 | Enable |
| S | RW | CH6LIMITH | | | Enable or disable interrupt for event CH6LIMITH |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| T | RW | CH6LIMITL | | | Enable or disable interrupt for event CH6LIMITL |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| U | RW | CH7LIMITH | | | Enable or disable interrupt for event CH7LIMITH |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| V | RW | CH7LIMITL | | | Enable or disable interrupt for event CH7LIMITL |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |

## 6.17.11.14 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | STARTED | | | Write '1' to enable interrupt for event STARTED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | END | | | Write '1' to enable interrupt for event END |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | DONE | | | Write '1' to enable interrupt for event DONE |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | RESULTDONE | | | Write '1' to enable interrupt for event RESULTDONE |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E | RW | CALIBRATEDONE | | | Write '1' to enable interrupt for event CALIBRATEDONE |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| F | RW | STOPPED | | | Write '1' to enable interrupt for event STOPPED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| G | RW | CH0LIMITH | | | Write '1' to enable interrupt for event CH0LIMITH |
| | | | Set | 1 | Enable |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| H | RW | CH0LIMITL | | | Write '1' to enable interrupt for event CH0LIMITL |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| I | RW | CH1LIMITH | | | Write '1' to enable interrupt for event CH1LIMITH |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| J | RW | CH1LIMITL | | | Write '1' to enable interrupt for event CH1LIMITL |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| K | RW | CH2LIMITH | | | Write '1' to enable interrupt for event CH2LIMITH |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| L | RW | CH2LIMITL | | | Write '1' to enable interrupt for event CH2LIMITL |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| M | RW | CH3LIMITH | | | Write '1' to enable interrupt for event CH3LIMITH |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| N | RW | CH3LIMITL | | | Write '1' to enable interrupt for event CH3LIMITL |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| O | RW | CH4LIMITH | | | Write '1' to enable interrupt for event CH4LIMITH |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| P | RW | CH4LIMITL | | | Write '1' to enable interrupt for event CH4LIMITL |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| Q | RW | CH5LIMITH | | | Write '1' to enable interrupt for event CH5LIMITH |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| R | RW | CH5LIMITL | | | Write '1' to enable interrupt for event CH5LIMITL |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| S | RW | CH6LIMITH | | | Write '1' to enable interrupt for event CH6LIMITH |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| T | RW | CH6LIMITL | | | Write '1' to enable interrupt for event CH6LIMITL |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| U | RW | CH7LIMITH | | | Write '1' to enable interrupt for event CH7LIMITH |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| V | RW | CH7LIMITL | | | Write '1' to enable interrupt for event CH7LIMITL |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.17.11.15 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | STARTED | | | Write '1' to disable interrupt for event STARTED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | END | | | Write '1' to disable interrupt for event END |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | DONE | | | Write '1' to disable interrupt for event DONE |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | RESULTDONE | | | Write '1' to disable interrupt for event RESULTDONE |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E | RW | CALIBRATEDONE | | | Write '1' to disable interrupt for event CALIBRATEDONE |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| F | RW | STOPPED | | | Write '1' to disable interrupt for event STOPPED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| G | RW | CH0LIMITH | | | Write '1' to disable interrupt for event CH0LIMITH |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| | | | Enabled | 1 | Read: Enabled |
| H | RW | CH0LIMITL | | | Write '1' to disable interrupt for event CH0LIMITL |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| I | RW | CH1LIMITH | | | Write '1' to disable interrupt for event CH1LIMITH |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| J | RW | CH1LIMITL | | | Write '1' to disable interrupt for event CH1LIMITL |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| K | RW | CH2LIMITH | | | Write '1' to disable interrupt for event CH2LIMITH |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| L | RW | CH2LIMITL | | | Write '1' to disable interrupt for event CH2LIMITL |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| M | RW | CH3LIMITH | | | Write '1' to disable interrupt for event CH3LIMITH |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| N | RW | CH3LIMITL | | | Write '1' to disable interrupt for event CH3LIMITL |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| O | RW | CH4LIMITH | | | Write '1' to disable interrupt for event CH4LIMITH |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| P | RW | CH4LIMITL | | | Write '1' to disable interrupt for event CH4LIMITL |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| Q | RW | CH5LIMITH | | | Write '1' to disable interrupt for event CH5LIMITH |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| R | RW | CH5LIMITL | | | Write '1' to disable interrupt for event CH5LIMITL |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| S | RW | CH6LIMITH | | | Write '1' to disable interrupt for event CH6LIMITH |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| T | RW | CH6LIMITL | | | Write '1' to disable interrupt for event CH6LIMITL |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | V U T S R Q P O N M L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| U | RW | CH7LIMITH | | | Write '1' to disable interrupt for event CH7LIMITH |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| V | RW | CH7LIMITL | | | Write '1' to disable interrupt for event CH7LIMITL |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.17.11.16 STATUS

Address offset: 0x400

Status

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | STATUS | | | Status |
| | | | Ready | 0 | ADC is ready. No on-going conversion. |
| | | | Busy | 1 | ADC is busy. Conversion in progress. |

## 6.17.11.17 ENABLE

Address offset: 0x500

Enable or disable ADC

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ENABLE | | | Enable or disable ADC |
| | | | Disabled | 0 | Disable ADC |
| | | | Enabled | 1 | Enable ADC |
| | | | | | When enabled, the ADC will acquire access to the analog input pins specified in the CH[n].PSELP and CH[n].PSELN registers. |

## 6.17.11.18 CH[n].PSELP (n=0..7)

Address offset: 0x510 + (n × 0x10)

Input positive pin selection for CH[n]

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | PSELP | | | Analog positive input channel |
| | | | NC | 0 | Not connected |
| | | | AnalogInput0 | 1 | AIN0 |
| | | | AnalogInput1 | 2 | AIN1 |
| | | | AnalogInput2 | 3 | AIN2 |
| | | | AnalogInput3 | 4 | AIN3 |
| | | | AnalogInput4 | 5 | AIN4 |
| | | | AnalogInput5 | 6 | AIN5 |
| | | | AnalogInput6 | 7 | AIN6 |
| | | | AnalogInput7 | 8 | AIN7 |
| | | | VDD | 9 | VDD |

## 6.17.11.19 CH[n].PSELN (n=0..7)

Address offset: 0x514 + (n × 0x10)

Input negative pin selection for CH[n]

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | PSELN | | | Analog negative input, enables differential channel |
| | | | NC | 0 | Not connected |
| | | | AnalogInput0 | 1 | AIN0 |
| | | | AnalogInput1 | 2 | AIN1 |
| | | | AnalogInput2 | 3 | AIN2 |
| | | | AnalogInput3 | 4 | AIN3 |
| | | | AnalogInput4 | 5 | AIN4 |
| | | | AnalogInput5 | 6 | AIN5 |
| | | | AnalogInput6 | 7 | AIN6 |
| | | | AnalogInput7 | 8 | AIN7 |
| | | | VDD | 9 | VDD |

## 6.17.11.20 CH[n].CONFIG (n=0..7)

Address offset: 0x518 + (n × 0x10)

Input configuration for CH[n]

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | G F E E E D C C C B B A A |
| **Reset 0x00020000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | RESP | | | Positive channel resistor control |
| | | | Bypass | 0 | Bypass resistor ladder |
| | | | Pulldown | 1 | Pull-down to GND |
| | | | Pullup | 2 | Pull-up to VDD |
| | | | VDD1_2 | 3 | Set input at VDD/2 |
| B | RW | RESN | | | Negative channel resistor control |
| | | | Bypass | 0 | Bypass resistor ladder |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | G F E E E D C C C B B A A |
| **Reset 0x00020000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| | | | Pulldown | 1 | Pull-down to GND |
| | | | Pullup | 2 | Pull-up to VDD |
| | | | VDD1_2 | 3 | Set input at VDD/2 |
| C | RW | GAIN | | | Gain control |
| | | | Gain1_6 | 0 | 1/6 |
| | | | Gain1_5 | 1 | 1/5 |
| | | | Gain1_4 | 2 | 1/4 |
| | | | Gain1_3 | 3 | 1/3 |
| | | | Gain1_2 | 4 | 1/2 |
| | | | Gain1 | 5 | 1 |
| | | | Gain2 | 6 | 2 |
| | | | Gain4 | 7 | 4 |
| D | RW | REFSEL | | | Reference control |
| | | | Internal | 0 | Internal reference (0.6 V) |
| | | | VDD1_4 | 1 | VDD/4 as reference |
| E | RW | TACQ | | | Acquisition time, the time the ADC uses to sample the input voltage |
| | | | 3us | 0 | 3 us |
| | | | 5us | 1 | 5 us |
| | | | 10us | 2 | 10 us |
| | | | 15us | 3 | 15 us |
| | | | 20us | 4 | 20 us |
| | | | 40us | 5 | 40 us |
| F | RW | MODE | | | Enable differential mode |
| | | | SE | 0 | Single ended, PSELN will be ignored, negative input to ADC shorted to GND |
| | | | Diff | 1 | Differential |
| G | RW | BURST | | | Enable burst mode |
| | | | Disabled | 0 | Burst mode is disabled (normal operation) |
| | | | Enabled | 1 | Burst mode is enabled. SAADC takes 2^OVERSAMPLE number of samples as fast as it can, and sends the average to Data RAM. |

## 6.17.11.21 CH[n].LIMIT (n=0..7)

Address offset: 0x51C + (n × 0x10)

High/low limits for event monitoring a channel

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | B B B B B B B B B B B B B B B B A A A A A A A A A A A A A A |
| **Reset 0x7FFF8000** | | | | **0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | LOW | | [-32768 to +32767] | Low level limit |
| B | RW | HIGH | | [-32768 to +32767] | High level limit |

## 6.17.11.22 RESOLUTION

Address offset: 0x5F0

Resolution configuration

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A |
| **Reset 0x00000001** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | VAL | | | Set the resolution |
| | | | 8bit | 0 | 8 bit |
| | | | 10bit | 1 | 10 bit |
| | | | 12bit | 2 | 12 bit |
| | | | 14bit | 3 | 14 bit |

## 6.17.11.23 OVERSAMPLE

Address offset: 0x5F4

Oversampling configuration. OVERSAMPLE should not be combined with SCAN. The RESOLUTION is applied before averaging, thus for high OVERSAMPLE a higher RESOLUTION should be used.

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | OVERSAMPLE | | | Oversample control |
| | | | Bypass | 0 | Bypass oversampling |
| | | | Over2x | 1 | Oversample 2x |
| | | | Over4x | 2 | Oversample 4x |
| | | | Over8x | 3 | Oversample 8x |
| | | | Over16x | 4 | Oversample 16x |
| | | | Over32x | 5 | Oversample 32x |
| | | | Over64x | 6 | Oversample 64x |
| | | | Over128x | 7 | Oversample 128x |
| | | | Over256x | 8 | Oversample 256x |

## 6.17.11.24 SAMPLERATE

Address offset: 0x5F8

Controls normal or continuous sample rate

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | B A A A A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | CC | | [80..2047] | Capture and compare value. Sample rate is 16 MHz/CC |
| B | RW | MODE | | | Select mode for sample rate control |
| | | | Task | 0 | Rate is controlled from SAMPLE task |
| | | | Timers | 1 | Rate is controlled from local timer (use CC to control the rate) |

## 6.17.11.25 RESULT.PTR

Address offset: 0x62C

Data pointer

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | PTR | | | Data pointer |

## 6.17.11.26 RESULT.MAXCNT

Address offset: 0x630

Maximum number of buffer words to transfer

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | MAXCNT | | | Maximum number of buffer words to transfer |

## 6.17.11.27 RESULT.AMOUNT

Address offset: 0x634

Number of buffer words transferred since last START

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | AMOUNT | | | Number of buffer words transferred since last START. This register can be read after an END or STOPPED event. |

# 6.17.12 Electrical specification

## 6.17.12.1 SAADC Electrical Specification

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $DNL_{10}$ | Differential non-linearity, 10-bit resolution | -0.95 | <1 | | LSB10b |
| $INL_{10}$ | Integral non-linearity, 10-bit resolution | | 1 | | LSB1( |
| $V_{OS}$ | Differential offset error (calibrated), 10-bit resolution [a] | | +-2 | | LSB10b |
| $DNL_{12}$ | Differential non-linearity, 12-bit resolution | -0.95 | 1.3 | | LSB1: |
| $INL_{12}$ | Integral non-linearity, 12-bit resolution | | 4.7 | | LSB12b |
| $C_{EG}$ | Gain error temperature coefficient | | 0.02 | | %/°C |
| $f_{SAMPLE}$ | Maximum sampling rate | | | 200 | kHz |
| $t_{ACQ,10k}$ | Acquisition time (configurable), source Resistance <= 10kOhm | | 3 | | µs |
| $t_{ACQ,40k}$ | Acquisition time (configurable), source Resistance <= 40kOhm | | 5 | | µs |
| $t_{ACQ,100k}$ | Acquisition time (configurable), source Resistance <= 100kOhm | | 10 | | µs |
| $t_{ACQ,200k}$ | Acquisition time (configurable), source Resistance <= 200kOhm | | 15 | | µs |

[a] Digital output code at zero volt differential input.

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{ACQ,400k}$ | Acquisition time (configurable), source Resistance <= 400kOhm | | 20 | | µs |
| $t_{ACQ,800k}$ | Acquisition time (configurable), source Resistance <= 800kOhm | | 40 | | µs |
| $t_{CONV}$ | Conversion time | | <2 | | µs |
| $E_{G1/6}$ | Error[b] for Gain = 1/6 | -3 | | 3 | % |
| $E_{G1/4}$ | Error[b] for Gain = 1/4 | -3 | | 3 | % |
| $E_{G1/2}$ | Error[b] for Gain = 1/2 | -3 | | 4 | % |
| $E_{G1}$ | Error[b] for Gain = 1 | -3 | | 4 | % |
| $C_{SAMPLE}$ | Sample and hold capacitance at maximum gain[21] | | 2.5 | | pF |
| $R_{INPUT}$ | Input resistance | | >1 | | MΩ |
| $E_{NOB}$ | Effective number of bits, differential mode, 12-bit resolution, 1/1 gain, 3 µs acquisition time, crystal HFCLK, 200 ksps | | 9 | | Bit |
| $S_{NDR}$ | Peak signal to noise and distortion ratio, differential mode, 12-bit resolution, 1/1 gain, 3 µs acquisition time, crystal HFCLK, 200 ksps | | 56 | | dB |
| $S_{FDR}$ | Spurious free dynamic range, differential mode, 12-bit resolution, 1/1 gain, 3 µs acquisition time, crystal HFCLK, 200 ksps | | 70 | | dBc |
| $R_{LADDER}$ | Ladder resistance | | 160 | | kΩ |

### 6.17.13 Performance factors

Clock jitter, affecting sample timing accuracy, and circuit noise can affect ADC performance.

Jitter can be between START tasks or from START task to acquisition. START timer accuracy and startup times of regulators and references will contribute to variability. Sources of circuit noise may include CPU activity and the DC/DC regulator. Best ADC performance is achieved using START timing based on the TIMER module, HFXO clock source, and Constant Latency mode.

## 6.18 SPI — Serial peripheral interface master

The SPI master provides a simple CPU interface which includes a TXD register for sending data and an RXD register for receiving data. This section is added for legacy support for now.

[b] Does not include temperature drift

[21] Maximum gain corresponds to highest capacitance.

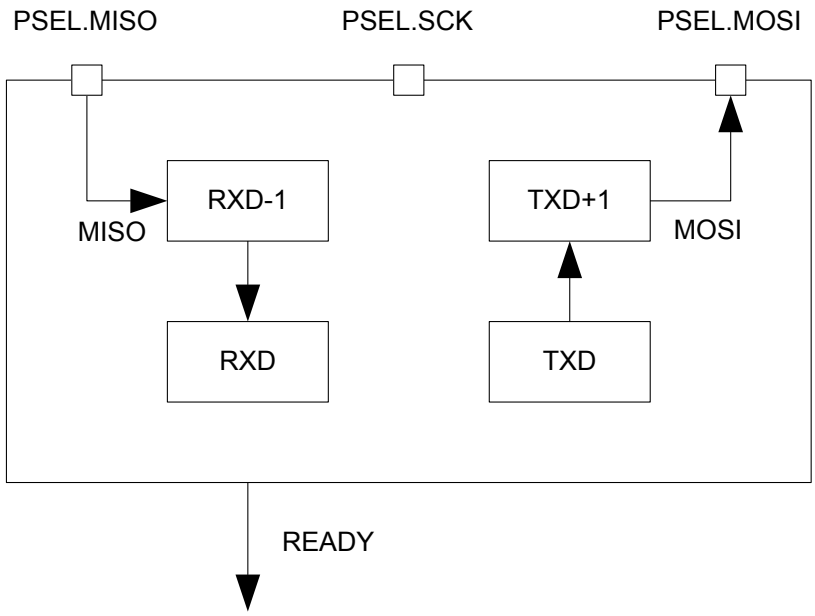



*Figure 111: SPI master*

RXD-1 and TXD+1 illustrate the double buffered version of RXD and TXD respectively.

## 6.18.1 Functional description

The TXD and RXD registers are double-buffered to enable some degree of uninterrupted data flow in and out of the SPI master.

The SPI master does not implement support for chip select directly. Therefore, the CPU must use available GPIOs to select the correct slave and control this independently of the SPI master. The SPI master supports SPI modes 0 through 3.

| Mode | Clock polarity | Clock phase |
|---|---|---|
| | CPOL | CPHA |
| SPI_MODE0 | 0 (Leading) | 0 (Active high) |
| SPI_MODE1 | 0 (Leading) | 1 (Active low) |
| SPI_MODE2 | 1 (Trailing) | 0 (Active high) |
| SPI_MODE3 | 1 (Trailing) | 1 (Active low) |

*Table 81: SPI modes*

### 6.18.1.1 SPI master mode pin configuration

The different signals SCK, MOSI, and MISO associated with the SPI master are mapped to physical pins.

This mapping is according to the configuration specified in the PSEL.SCK, PSEL.MOSI, and PSEL.MISO registers respectively. If the CONNECT field of a PSEL.xxx register is set to Disconnected, the associated SPI master signal is not connected to any physical pin. The PSEL.SCK, PSEL.MOSI, and PSEL.MISO registers and their configurations are only used as long as the SPI master is enabled, and retained only as long as the device is in ON mode. PSEL.SCK, PSEL.MOSI, and PSEL.MISO must only be configured when the SPI master is disabled.

To secure correct behavior in the SPI, the pins used by the SPI must be configured in the GPIO peripheral as described in GPIO configuration on page 284 prior to enabling the SPI. The SCK must always be connected to a pin, and that pin's input buffer must always be connected for the SPI to work. This configuration must be retained in the GPIO for the selected IOs as long as the SPI is enabled.

Only one peripheral can be assigned to drive a particular GPIO pin at a time, failing to do so may result in unpredictable behavior.

| SPI master signal | SPI master pin | Direction | Output value |
|---|---|---|---|
| SCK | As specified in PSEL.SCK | Output | Same as CONFIG.CPOL |
| MOSI | As specified in PSEL.MOSI | Output | 0 |
| MISO | As specified in PSEL.MISO | Input | Not applicable |

*Table 82: GPIO configuration*

### 6.18.1.2 Shared resources

The SPI shares registers and other resources with other peripherals that have the same ID as the SPI. Therefore, the user must disable all peripherals that have the same ID as the SPI before the SPI can be configured and used.

Disabling a peripheral that has the same ID as the SPI will not reset any of the registers that are shared with the SPI. It is therefore important to configure all relevant SPI registers explicitly to secure that it operates correctly.

See the Instantiation table in Instantiation on page 18 for details on peripherals and their IDs.

### 6.18.1.3 SPI master transaction sequence

An SPI master transaction is started by writing the first byte, which is to be transmitted by the SPI master, to the TXD register.

Since the transmitter is double buffered, the second byte can be written to the TXD register immediately after the first one. The SPI master will then send these bytes in the order they are written to the TXD register.

The SPI master is a synchronous interface, and for every byte that is sent, a different byte will be received at the same time; this is illustrated in SPI master transaction on page 285. Bytes that are received will be moved to the RXD register where the CPU can extract them by reading the register. The RXD register is double buffered in the same way as the TXD register, and a second byte can therefore be received at the same time as the first byte is being extracted from RXD by the CPU. The SPI master will generate a READY event every time a new byte is moved to the RXD register. The double buffered byte will be moved from RXD-1 to RXD as soon as the first byte is extracted from RXD. The SPI master will stop when there are no more bytes to send in TXD and TXD+1.

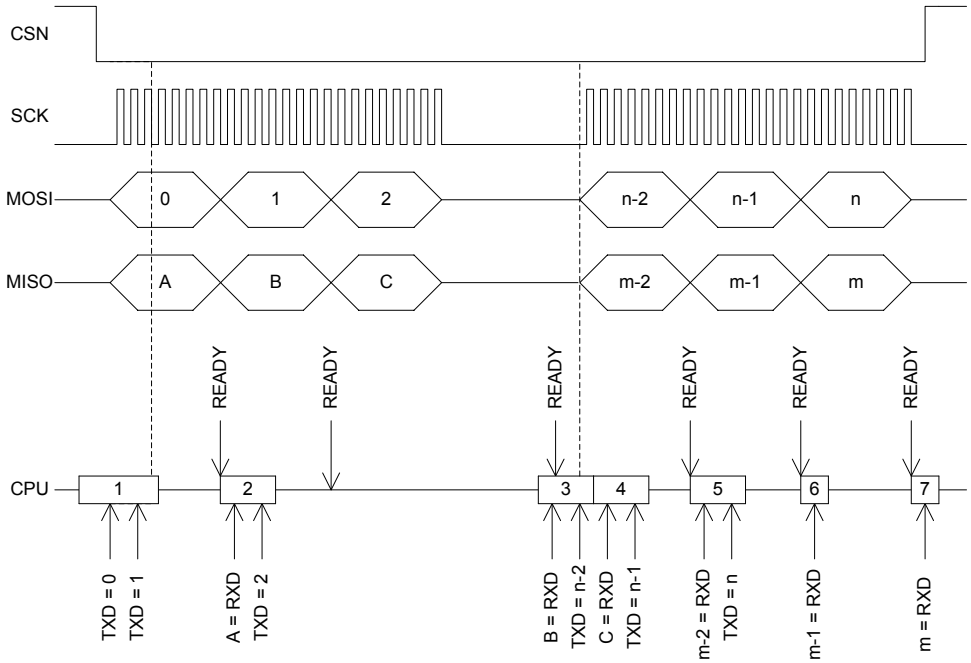



*Figure 112: SPI master transaction*

The READY event of the third byte transaction is delayed until B is extracted from RXD in occurrence number 3 on the horizontal lifeline. The reason for this is that the third event is generated first when C is moved from RXD-1 to RXD after B is read.

The SPI master will move the incoming byte to the RXD register after a short delay following the SCK clock period of the last bit in the byte. This also means that the READY event will be delayed accordingly, see SPI master transaction on page 285. Therefore, it is important that you always clear the READY event, even if the RXD register and the data that is being received is not used.

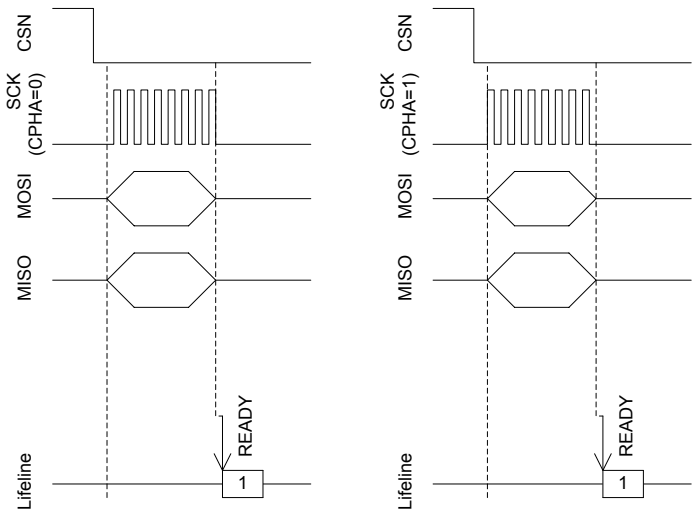



*Figure 113: SPI master transaction*

## 6.18.2 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40003000 | SPI | SPI1 | SPI master 1 | Deprecated |
| 0x40004000 | SPI | SPI0 | SPI master 0 | Deprecated |

*Table 83: Instances*

| Register | Offset | Description |
|---|---|---|
| EVENTS_READY | 0x108 | TXD byte sent and RXD byte received |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| ENABLE | 0x500 | Enable SPI |
| PSEL.SCK | 0x508 | Pin select for SCK |
| PSEL.MOSI | 0x50C | Pin select for MOSI signal |
| PSEL.MISO | 0x510 | Pin select for MISO signal |
| RXD | 0x518 | RXD register |
| TXD | 0x51C | TXD register |
| FREQUENCY | 0x524 | SPI frequency. Accuracy depends on the HFCLK source selected. |
| CONFIG | 0x554 | Configuration register |

*Table 84: Register overview*

### 6.18.2.1 EVENTS_READY

Address offset: 0x108

TXD byte sent and RXD byte received

| Bit number | | | | | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | A |
| **Reset 0x00000000** | | | | | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| ID | Acce | Field | Value ID | Value | Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| A | RW | EVENTS_READY | | | TXD byte sent and RXD byte received | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | NotGenerated | 0 | Event not generated | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | Generated | 1 | Event generated | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

### 6.18.2.2 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | A | | |
| **Reset 0x00000000** | | | | | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| ID | Acce | Field | Value ID | Value | Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| A | RW | READY | | | Write '1' to enable interrupt for event READY | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | Set | 1 | Enable | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | Disabled | 0 | Read: Disabled | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | Enabled | 1 | Read: Enabled | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

### 6.18.2.3 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | READY | | | Write '1' to disable interrupt for event READY |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

### 6.18.2.4 ENABLE

Address offset: 0x500

Enable SPI

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ENABLE | | | Enable or disable SPI |
| | | | Disabled | 0 | Disable SPI |
| | | | Enabled | 1 | Enable SPI |

### 6.18.2.5 PSEL.SCK

Address offset: 0x508

Pin select for SCK

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

### 6.18.2.6 PSEL.MOSI

Address offset: 0x50C

Pin select for MOSI signal

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.18.2.7 PSEL.MISO

Address offset: 0x510

Pin select for MISO signal

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.18.2.8 RXD

Address offset: 0x518

RXD register

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | RXD | | | RX data received. Double buffered |

## 6.18.2.9 TXD

Address offset: 0x51C

TXD register

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | TXD | | | TX data to send. Double buffered |

## 6.18.2.10 FREQUENCY

Address offset: 0x524

SPI frequency. Accuracy depends on the HFCLK source selected.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x04000000** | | | | **0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | FREQUENCY | | | SPI master data rate |
| | | | K125 | 0x02000000 | 125 kbps |
| | | | K250 | 0x04000000 | 250 kbps |
| | | | K500 | 0x08000000 | 500 kbps |
| | | | M1 | 0x10000000 | 1 Mbps |
| | | | M2 | 0x20000000 | 2 Mbps |
| | | | M4 | 0x40000000 | 4 Mbps |
| | | | M8 | 0x80000000 | 8 Mbps |

### 6.18.2.11 CONFIG

Address offset: 0x554

Configuration register

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ORDER | | | Bit order |
| | | | MsbFirst | 0 | Most significant bit shifted out first |
| | | | LsbFirst | 1 | Least significant bit shifted out first |
| B | RW | CPHA | | | Serial clock (SCK) phase |
| | | | Leading | 0 | Sample on leading edge of clock, shift serial data on trailing edge |
| | | | Trailing | 1 | Sample on trailing edge of clock, shift serial data on leading edge |
| C | RW | CPOL | | | Serial clock (SCK) polarity |
| | | | ActiveHigh | 0 | Active high |
| | | | ActiveLow | 1 | Active low |

## 6.18.3 Electrical specification

### 6.18.3.1 SPI master interface electrical specifications

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $f_{SPI}$ | Bit rates for SPI[22] | | | 8[23] | Mbps |
| $t_{SPI,START}$ | Time from writing TXD register to transmission started | | 1 | | µs |

### 6.18.3.2 Serial Peripheral Interface (SPI) Master timing specifications

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{SPI,CSCK}$ | SCK period | 125 | | | ns |
| $t_{SPI,RSCK,LD}$ | SCK rise time, standard drive[a] | | | $t_{RF,25pF}$ | |

22 High bit rates may require GPIOs to be set as High Drive, see GPIO chapter for more details.

23 The actual maximum data rate depends on the slave's CLK to MISO and MOSI setup and hold timings.

a At 25pF load, including GPIO capacitance, see GPIO spec.

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{SPI,RSCK,HD}$ | SCK rise time, high drive[a] | | | $t_{HRF,25pF}$ | |
| $t_{SPI,FSCK,LD}$ | SCK fall time, standard drive[a] | | | $t_{RF,25pF}$ | |
| $t_{SPI,FSCK,HD}$ | SCK fall time, high drive[a] | | | $t_{HRF,25pF}$ | |
| $t_{SPI,WHSCK}$ | SCK high time[a] | $(t_{CSCK}/2) - t_{RSCK}$ | | | |
| $t_{SPI,WLSCK}$ | SCK low time[a] | $(t_{CSCK}/2) - t_{FSCK}$ | | | |
| $t_{SPI,SUMI}$ | MISO to CLK edge setup time | 19 | | | ns |
| $t_{SPI,HMI}$ | CLK edge to MISO hold time | 18 | | | ns |
| $t_{SPI,VMO}$ | CLK edge to MOSI valid | | | 59 | ns |
| $t_{SPI,HMO}$ | MOSI hold time after CLK edge | 20 | | | ns |

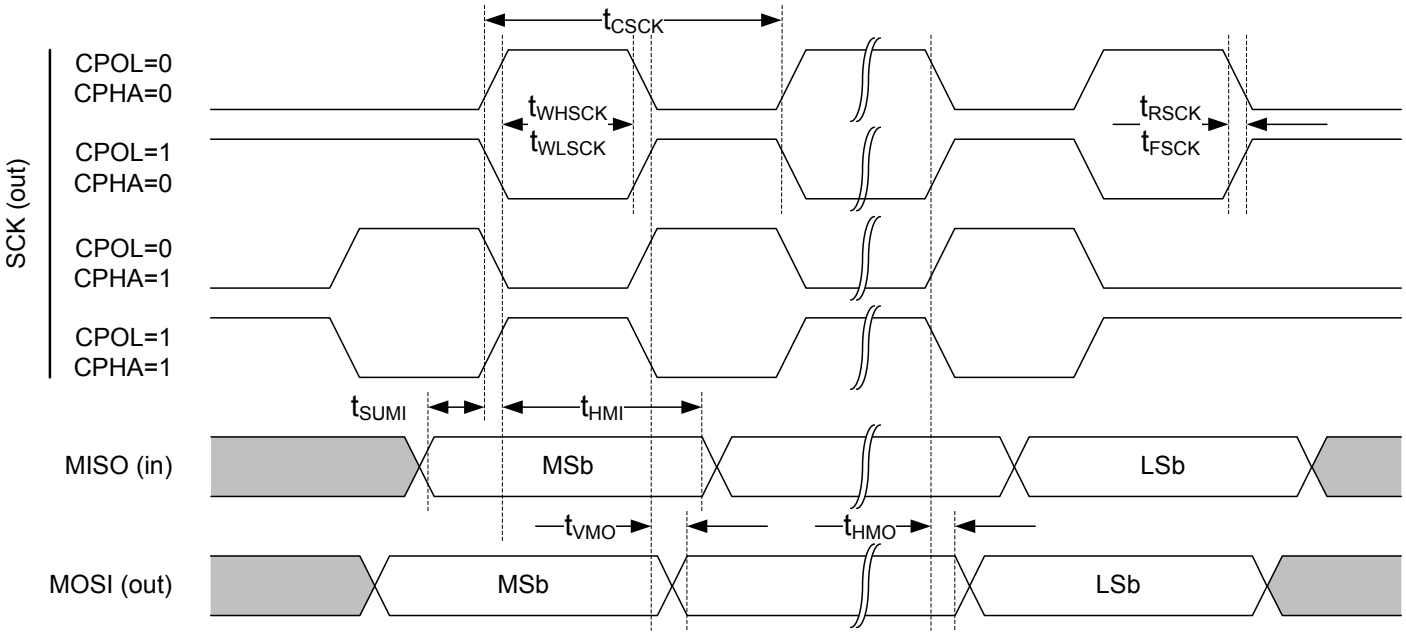



*Figure 114: SPI master timing diagram*

# 6.19 SPIM — Serial peripheral interface master with EasyDMA

The SPI master can communicate with multiple slaves using individual chip select signals for each of the slave devices attached to a bus.

Listed here are the main features for the SPIM

- SPI mode 0-3
- EasyDMA direct transfer to/from RAM for both SPI Slave and SPI Master
- Individual selection of IO pin for each SPI signal

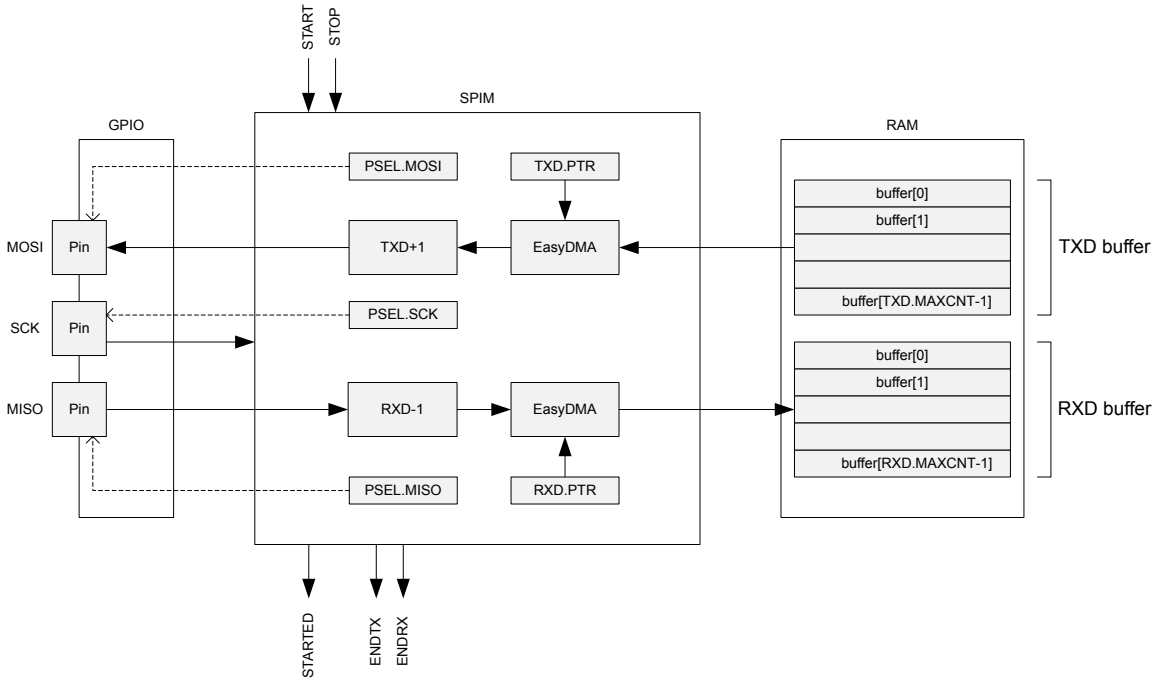



*Figure 115: SPIM — SPI master with EasyDMA*

The SPIM does not implement support for chip select directly. Therefore, the CPU must use available GPIOs to select the correct slave and control this independently of the SPI master. The SPIM supports SPI modes 0 through 3. The CONFIG register allows setting CPOL and CPHA appropriately.

| Mode | Clock polarity | Clock phase |
|---|---|---|
| | CPOL | CPHA |
| SPI_MODE0 | 0 (Active High) | 0 (Leading) |
| SPI_MODE1 | 0 (Active High) | 1 (Trailing) |
| SPI_MODE2 | 1 (Active Low) | 0 (Leading) |
| SPI_MODE3 | 1 (Active Low) | 1 (Trailing) |

*Table 85: SPI modes*

## 6.19.1 SPI master transaction sequence

An SPI master transaction consists of a sequence started by the START task followed by a number of events, and finally the STOP task.

An SPI master transaction is started by triggering the START task. The ENDTX event will be generated when the transmitter has transmitted all bytes in the TXD buffer as specified in the TXD.MAXCNT register. The ENDRX event will be generated when the receiver has filled the RXD buffer, i.e. received the last possible byte as specified in the RXD.MAXCNT register.

Following a START task, the SPI master will generate an END event when both ENDRX and ENDTX have been generated.

The SPI master is stopped by triggering the STOP task. A STOPPED event is generated when the SPI master has stopped.

If the ENDRX event has not already been generated when the SPI master has come to a stop, the SPI master will generate the ENDRX event explicitly even though the RX buffer is not full.

If the ENDTX event has not already been generated when the SPI master has come to a stop, the SPI master will generate the ENDTX event explicitly even though all bytes in the TXD buffer, as specified in the TXD.MAXCNT register, have not been transmitted.

The SPI master is a synchronous interface, and for every byte that is sent, a different byte will be received at the same time; this is illustrated in SPI master transaction on page 292.

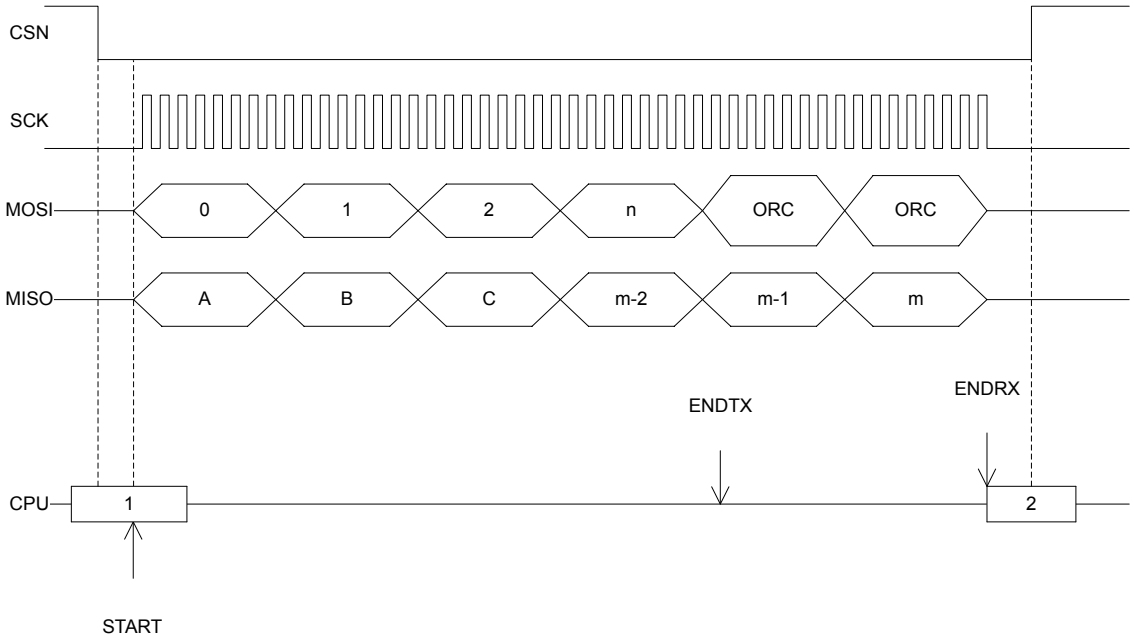



*Figure 116: SPI master transaction*

## 6.19.2 Master mode pin configuration

The SCK, MOSI, and MISO signals associated with the SPI master are mapped to physical pins according to the configuration specified in the PSEL.SCK, PSEL.MOSI, and PSEL.MISO registers respectively.

The PSEL.SCK, PSEL.MOSI, and PSEL.MISO registers and their configurations are only used as long as the SPI master is enabled, and retained only as long as the device is in ON mode. PSEL.SCK, PSEL.MOSI and PSEL.MISO must only be configured when the SPI master is disabled.

To secure correct behavior in the SPI, the pins used by the SPI must be configured in the GPIO peripheral as described in GPIO configuration on page 292 prior to enabling the SPI. This configuration must be retained in the GPIO for the selected IOs as long as the SPI is enabled.

Only one peripheral can be assigned to drive a particular GPIO pin at a time. Failing to do so may result in unpredictable behavior.

| SPI master signal | SPI master pin | Direction | Output value |
|---|---|---|---|
| SCK | As specified in PSEL.SCK | Output | Same as CONFIG.CPOL |
| MOSI | As specified in PSEL.MOSI | Output | 0 |
| MISO | As specified in PSEL.MISO | Input | Not applicable |

*Table 86: GPIO configuration*

## 6.19.3 EasyDMA

The SPIM implements EasyDMA for accessing RAM without CPU involvement.

The SPIM peripheral implements the following EasyDMA channels:

| Channel | Type | Register Cluster |
|---|---|---|
| TXD | READER | TXD |
| RXD | WRITER | RXD |

*Table 87: SPIM EasyDMA Channels*

For detailed information regarding the use of EasyDMA, see EasyDMA on page 35.

The .PTR and .MAXCNT registers are double-buffered. They can be updated and prepared for the next transmission immediately after having received the STARTED event.

The SPI master will automatically stop transmitting after TXD.MAXCNT bytes have been transmitted and RXD.MAXCNT bytes have been received. If RXD.MAXCNT is larger than TXD.MAXCNT, the remaining transmitted bytes will contain the value defined in the ORC register. If TXD.MAXCNT is larger than RXD.MAXCNT, the superfluous received bytes will be discarded.

The ENDRX/ENDTX event indicate that EasyDMA has finished accessing respectively the RX/TX buffer in RAM. The END event gets generated when both RX and TX are finished accessing the buffers in RAM.

In the case of bus congestion as described in , data loss may occur.

## 6.19.4 Low power

When putting the system in low power and the peripheral is not needed, lowest possible power consumption is achieved by stopping, and then disabling the peripheral.

The STOP task may not be always needed (the peripheral might already be stopped), but if it is sent, software shall wait until the STOPPED event was received as a response before disabling the peripheral through the ENABLE register.

## 6.19.5 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40003000 | SPIM | SPIM1 | SPI master 1 | |
| 0x40004000 | SPIM | SPIM0 | SPI master 0 | |

*Table 88: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_START | 0x010 | Start SPI transaction |
| TASKS_STOP | 0x014 | Stop SPI transaction |
| TASKS_SUSPEND | 0x01C | Suspend SPI transaction |
| TASKS_RESUME | 0x020 | Resume SPI transaction |
| EVENTS_STOPPED | 0x104 | SPI transaction has stopped |
| EVENTS_ENDRX | 0x110 | End of RXD buffer reached |
| EVENTS_END | 0x118 | End of RXD buffer and TXD buffer reached |
| EVENTS_ENDTX | 0x120 | End of TXD buffer reached |
| EVENTS_STARTED | 0x14C | Transaction started |
| SHORTS | 0x200 | Shortcuts between local events and tasks |

| Register | Offset | Description |
|---|---|---|
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| ENABLE | 0x500 | Enable SPIM |
| PSEL.SCK | 0x508 | Pin select for SCK |
| PSEL.MOSI | 0x50C | Pin select for MOSI signal |
| PSEL.MISO | 0x510 | Pin select for MISO signal |
| FREQUENCY | 0x524 | SPI frequency. Accuracy depends on the HFCLK source selected. |
| RXD.PTR | 0x534 | Data pointer |
| RXD.MAXCNT | 0x538 | Maximum number of bytes in receive buffer |
| RXD.AMOUNT | 0x53C | Number of bytes transferred in the last transaction |
| RXD.LIST | 0x540 | EasyDMA list type |
| TXD.PTR | 0x544 | Data pointer |
| TXD.MAXCNT | 0x548 | Maximum number of bytes in transmit buffer |
| TXD.AMOUNT | 0x54C | Number of bytes transferred in the last transaction |
| TXD.LIST | 0x550 | EasyDMA list type |
| CONFIG | 0x554 | Configuration register |
| ORC | 0x5C0 | Over-read character. Character clocked out in case and over-read of the TXD buffer. |

*Table 89: Register overview*

### 6.19.5.1 TASKS_START

Address offset: 0x010

Start SPI transaction

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_START | | | Start SPI transaction |
| | | | Trigger | 1 | Trigger task |

### 6.19.5.2 TASKS_STOP

Address offset: 0x014

Stop SPI transaction

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STOP | | | Stop SPI transaction |
| | | | Trigger | 1 | Trigger task |

### 6.19.5.3 TASKS_SUSPEND

Address offset: 0x01C

Suspend SPI transaction

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_SUSPEND | | | Suspend SPI transaction |
| | | | Trigger | 1 | Trigger task |

### 6.19.5.4 TASKS_RESUME

Address offset: 0x020

Resume SPI transaction

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_RESUME | | | Resume SPI transaction |
| | | | Trigger | 1 | Trigger task |

### 6.19.5.5 EVENTS_STOPPED

Address offset: 0x104

SPI transaction has stopped

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_STOPPED | | | SPI transaction has stopped |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.19.5.6 EVENTS_ENDRX

Address offset: 0x110

End of RXD buffer reached

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_ENDRX | | | End of RXD buffer reached |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.19.5.7 EVENTS_END

Address offset: 0x118

End of RXD buffer and TXD buffer reached

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_END | | | End of RXD buffer and TXD buffer reached |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.19.5.8 EVENTS_ENDTX

Address offset: 0x120

End of TXD buffer reached

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_ENDTX | | | End of TXD buffer reached |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.19.5.9 EVENTS_STARTED

Address offset: 0x14C

Transaction started

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_STARTED | | | Transaction started |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.19.5.10 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | END_START | | | Shortcut between event END and task START |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |

### 6.19.5.11 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | STOPPED | | | Write '1' to enable interrupt for event STOPPED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | ENDRX | | | Write '1' to enable interrupt for event ENDRX |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | END | | | Write '1' to enable interrupt for event END |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | ENDTX | | | Write '1' to enable interrupt for event ENDTX |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E | RW | STARTED | | | Write '1' to enable interrupt for event STARTED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

### 6.19.5.12 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | STOPPED | | | Write '1' to disable interrupt for event STOPPED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | ENDRX | | | Write '1' to disable interrupt for event ENDRX |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | END | | | Write '1' to disable interrupt for event END |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | ENDTX | | | Write '1' to disable interrupt for event ENDTX |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E | RW | STARTED | | | Write '1' to disable interrupt for event STARTED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | E D C B A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| | | | Enabled | 1 | Read: Enabled |

## 6.19.5.13 ENABLE

Address offset: 0x500

Enable SPIM

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ENABLE | | | Enable or disable SPIM |
| | | | Disabled | 0 | Disable SPIM |
| | | | Enabled | 7 | Enable SPIM |

## 6.19.5.14 PSEL.SCK

Address offset: 0x508

Pin select for SCK

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.19.5.15 PSEL.MOSI

Address offset: 0x50C

Pin select for MOSI signal

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.19.5.16 PSEL.MISO

Address offset: 0x510

Pin select for MISO signal

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.19.5.17 FREQUENCY

Address offset: 0x524

SPI frequency. Accuracy depends on the HFCLK source selected.

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x04000000** | | | **0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | FREQUENCY | | | SPI master data rate |
| | | | K125 | 0x02000000 | 125 kbps |
| | | | K250 | 0x04000000 | 250 kbps |
| | | | K500 | 0x08000000 | 500 kbps |
| | | | M1 | 0x10000000 | 1 Mbps |
| | | | M2 | 0x20000000 | 2 Mbps |
| | | | M4 | 0x40000000 | 4 Mbps |
| | | | M8 | 0x80000000 | 8 Mbps |

## 6.19.5.18 RXD.PTR

Address offset: 0x534

Data pointer

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | PTR | | | Data pointer |

**Note:** See the memory chapter for details about which memories are available for EasyDMA.

## 6.19.5.19 RXD.MAXCNT

Address offset: 0x538

Maximum number of bytes in receive buffer

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | MAXCNT | | [1..0x3FFF] | Maximum number of bytes in receive buffer |

## 6.19.5.20 RXD.AMOUNT

Address offset: 0x53C

Number of bytes transferred in the last transaction

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | AMOUNT | | [1..0x3FFF] | Number of bytes transferred in the last transaction |

## 6.19.5.21 RXD.LIST

Address offset: 0x540

EasyDMA list type

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | LIST | | | List type |
| | | | Disabled | 0 | Disable EasyDMA list |
| | | | ArrayList | 1 | Use array list |

## 6.19.5.22 TXD.PTR

Address offset: 0x544

Data pointer

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PTR | | | Data pointer |

**Note:** See the memory chapter for details about which memories are available for EasyDMA.

## 6.19.5.23 TXD.MAXCNT

Address offset: 0x548

Maximum number of bytes in transmit buffer

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | MAXCNT | | [1..0x3FFF] | Maximum number of bytes in transmit buffer |

## 6.19.5.24 TXD.AMOUNT

Address offset: 0x54C

Number of bytes transferred in the last transaction

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | AMOUNT | | [1..0x3FFF] | Number of bytes transferred in the last transaction |

## 6.19.5.25 TXD.LIST

Address offset: 0x550

EasyDMA list type

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | LIST | | | List type |
| | | | Disabled | 0 | Disable EasyDMA list |
| | | | ArrayList | 1 | Use array list |

## 6.19.5.26 CONFIG

Address offset: 0x554

Configuration register

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ORDER | | | Bit order |
| | | | MsbFirst | 0 | Most significant bit shifted out first |
| | | | LsbFirst | 1 | Least significant bit shifted out first |
| B | RW | CPHA | | | Serial clock (SCK) phase |
| | | | Leading | 0 | Sample on leading edge of clock, shift serial data on trailing edge |
| | | | Trailing | 1 | Sample on trailing edge of clock, shift serial data on leading edge |
| C | RW | CPOL | | | Serial clock (SCK) polarity |
| | | | ActiveHigh | 0 | Active high |
| | | | ActiveLow | 1 | Active low |

## 6.19.5.27 ORC

Address offset: 0x5C0

Over-read character. Character clocked out in case and over-read of the TXD buffer.

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | ORC | | | Over-read character. Character clocked out in case and over-read of the TXD buffer. |

## 6.19.6 Electrical specification

### 6.19.6.1 SPIM master interface electrical specifications

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $f_{SPIM}$ | Bit rates for SPIM[24] | | | 8[25] | Mbps |
| $t_{SPIM,START}$ | Time from START task to transmission started | .. | .. | .. | µs |

### 6.19.6.2 Serial Peripheral Interface Master (SPIM) timing specifications

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{SPIM,CSCK}$ | SCK period | .. | .. | .. | ns |
| $t_{SPIM,RSCK,LD}$ | SCK rise time, standard drive[a] | | | $t_{RF,25pF}$ | |
| $t_{SPIM,RSCK,HD}$ | SCK rise time, high drive[a] | | | $t_{HRF,25pF}$ | |
| $t_{SPIM,FSCK,LD}$ | SCK fall time, standard drive[a] | | | $t_{RF,25pF}$ | |
| $t_{SPIM,FSCK,HD}$ | SCK fall time, high drive[a] | | | $t_{HRF,25pF}$ | |
| $t_{SPIM,WHSCK}$ | SCK high time[a] | $(0.5*t_{CSCK}) - t_{RSCK}$ | | | |
| $t_{SPIM,WLSCK}$ | SCK low time[a] | $(0.5*t_{CSCK}) - t_{FSCK}$ | | | |
| $t_{SPIM,SUMI}$ | MISO to CLK edge setup time | 19 | | | ns |
| $t_{SPIM,HMI}$ | CLK edge to MISO hold time | 18 | | | ns |
| $t_{SPIM,VMO}$ | CLK edge to MOSI valid | | | 59 | ns |
| $t_{SPIM,HMO}$ | MOSI hold time after CLK edge | 20 | | | ns |

[24] High bit rates may require GPIOs to be set as High Drive, see GPIO chapter for more details.
[25] The actual maximum data rate depends on the slave's CLK to MISO and MOSI setup and hold timings.
[a] At 25pF load, including GPIO pin capacitance, see GPIO spec.

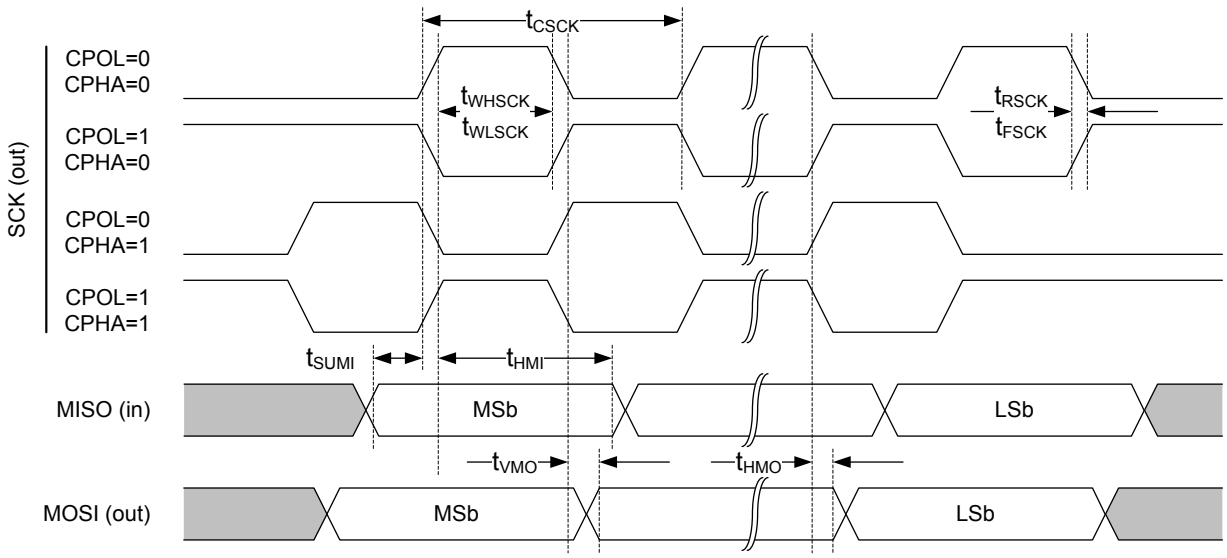



*Figure 117: SPIM timing diagram*

# 6.20 SPIS — Serial peripheral interface slave with EasyDMA

SPI slave (SPIS) is implemented with EasyDMA support for ultra low power serial communication from an external SPI master. EasyDMA in conjunction with hardware-based semaphore mechanisms removes all real-time requirements associated with controlling the SPI slave from a low priority CPU execution context.

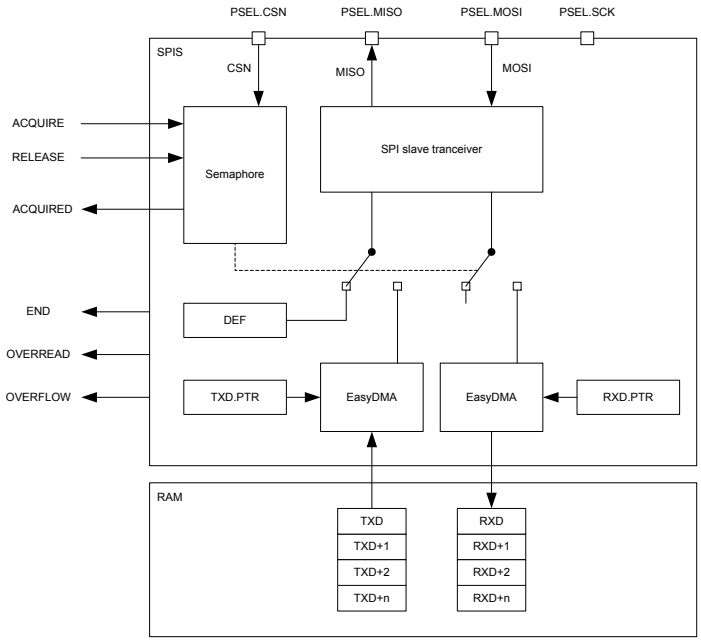



*Figure 118: SPI slave*

The SPIS supports SPI modes 0 through 3. The CONFIG register allows setting CPOL and CPHA appropriately.

| Mode | Clock polarity | Clock phase |
|---|---|---|
| | CPOL | CPHA |
| SPI_MODE0 | 0 (Active High) | 0 (Trailing Edge) |
| SPI_MODE1 | 0 (Active High) | 1 (Leading Edge) |
| SPI_MODE2 | 1 (Active Low) | 0 (Trailing Edge) |
| SPI_MODE3 | 1 (Active Low) | 1 (Leading Edge) |

*Table 90: SPI modes*

## 6.20.1 Shared resources

The SPI slave shares registers and other resources with other peripherals that have the same ID as the SPI slave. Therefore, you must disable all peripherals that have the same ID as the SPI slave before the SPI slave can be configured and used.

Disabling a peripheral that has the same ID as the SPI slave will not reset any of the registers that are shared with the SPI slave. It is important to configure all relevant SPI slave registers explicitly to secure that it operates correctly.

The Instantiation table in Instantiation on page 18 shows which peripherals have the same ID as the SPI slave.

## 6.20.2 EasyDMA

The SPIS implements EasyDMA for accessing RAM without CPU involvement.

The SPIS peripheral implements the following EasyDMA channels:

| Channel | Type | Register Cluster |
|---|---|---|
| TXD | READER | TXD |
| RXD | WRITER | RXD |

*Table 91: SPIS EasyDMA Channels*

For detailed information regarding the use of EasyDMA, see EasyDMA on page 35.

If RXD.MAXCNT is larger than TXD.MAXCNT, the remaining transmitted bytes will contain the value defined in the ORC register.

The END event indicates that EasyDMA has finished accessing the buffer in RAM.

## 6.20.3 SPI slave operation

SPI slave uses two memory pointers, RXD.PTR and TXD.PTR, that point to the RXD buffer (receive buffer) and TXD buffer (transmit buffer) respectively. Since these buffers are located in RAM, which can be accessed by both the SPI slave and the CPU, a hardware based semaphore mechanism is implemented to enable safe sharing.

See SPI transaction when shortcut between END and ACQUIRE is enabled on page 306.

Before the CPU can safely update the RXD.PTR and TXD.PTR pointers it must first acquire the SPI semaphore. The CPU can acquire the semaphore by triggering the ACQUIRE task and then receiving the ACQUIRED event. When the CPU has updated the RXD.PTR and TXD.PTR pointers the CPU must release the semaphore before the SPI slave will be able to acquire it. The CPU releases the semaphore by triggering the RELEASE task. This is illustrated in SPI transaction when shortcut between END and ACQUIRE is enabled on page 306. Triggering the RELEASE task when the semaphore is not granted to the CPU will have no effect.

The semaphore mechanism does not, at any time, prevent the CPU from performing read or write access to the RXD.PTR register, the TXD.PTR registers, or the RAM that these pointers are pointing to. The semaphore is only telling when these can be updated by the CPU so that safe sharing is achieved.

The semaphore is by default assigned to the CPU after the SPI slave is enabled. No ACQUIRED event will be generated for this initial semaphore handover. An ACQUIRED event will be generated immediately if the ACQUIRE task is triggered while the semaphore is assigned to the CPU.

The SPI slave will try to acquire the semaphore when CSN goes low. If the SPI slave does not manage to acquire the semaphore at this point, the transaction will be ignored. This means that all incoming data on MOSI will be discarded, and the DEF (default) character will be clocked out on the MISO line throughout the whole transaction. This will also be the case even if the semaphore is released by the CPU during the transaction. In case of a race condition where the CPU and the SPI slave try to acquire the semaphore at the same time, as illustrated in lifeline item 2 in SPI transaction when shortcut between END and ACQUIRE is enabled on page 306, the semaphore will be granted to the CPU.

If the SPI slave acquires the semaphore, the transaction will be granted. The incoming data on MOSI will be stored in the RXD buffer and the data in the TXD buffer will be clocked out on MISO.

When a granted transaction is completed and CSN goes high, the SPI slave will automatically release the semaphore and generate the END event.

As long as the semaphore is available the SPI slave can be granted multiple transactions one after the other. If the CPU is not able to reconfigure the TXD.PTR and RXD.PTR between granted transactions, the same TX data will be clocked out and the RX buffers will be overwritten. To prevent this from happening, the END_ACQUIRE shortcut can be used. With this shortcut enabled the semaphore will be handed over to the CPU automatically after the granted transaction has completed, giving the CPU the ability to update the TXPTR and RXPTR between every granted transaction.

If the CPU tries to acquire the semaphore while it is assigned to the SPI slave, an immediate handover will not be granted. However, the semaphore will be handed over to the CPU as soon as the SPI slave has released the semaphore after the granted transaction is completed. If the END_ACQUIRE shortcut is enabled and the CPU has triggered the ACQUIRE task during a granted transaction, only one ACQUIRE request will be served following the END event.

The MAXRX register specifies the maximum number of bytes the SPI slave can receive in one granted transaction. If the SPI slave receives more than MAXRX number of bytes, an OVERFLOW will be indicated in the STATUS register and the incoming bytes will be discarded.

The MAXTX parameter specifies the maximum number of bytes the SPI slave can transmit in one granted transaction. If the SPI slave is forced to transmit more than MAXTX number of bytes, an OVERREAD will be indicated in the STATUS register and the ORC character will be clocked out.

The RXD.AMOUNT and TXD.AMOUNT registers are updated when a granted transaction is completed. The TXD.AMOUNT register indicates how many bytes were read from the TX buffer in the last transaction, that is, ORC (over-read) characters are not included in this number. Similarly, the RXD.AMOUNT register indicates how many bytes were written into the RX buffer in the last transaction.

The ENDRX event is generated when the RX buffer has been filled.

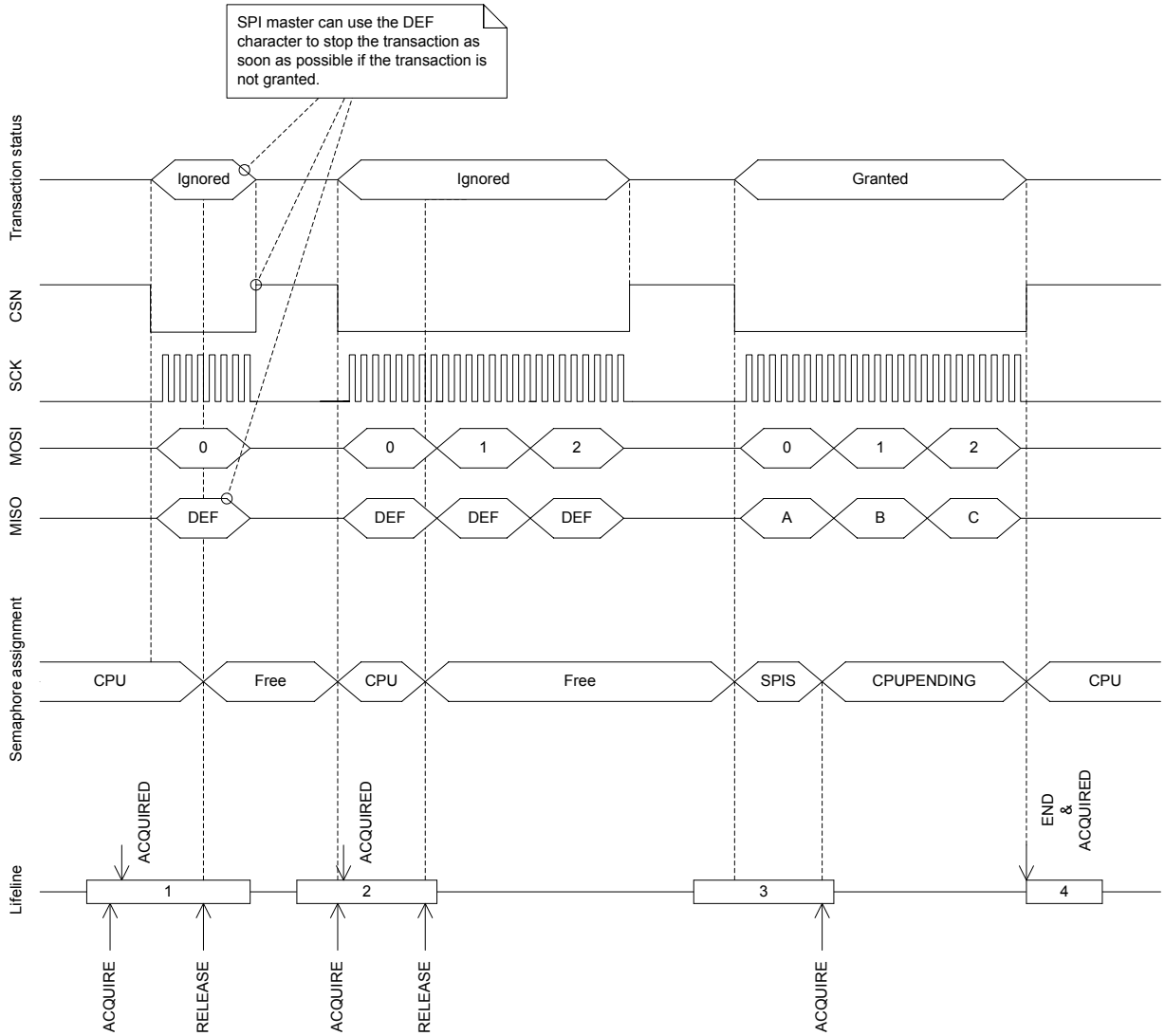



*Figure 119: SPI transaction when shortcut between END and ACQUIRE is enabled*

## 6.20.4 Pin configuration

The CSN, SCK, MOSI, and MISO signals associated with the SPI slave are mapped to physical pins according to the configuration specified in the PSEL.CSN, PSEL.SCK, PSEL.MOSI, and PSEL.MISO registers respectively. If the CONNECT field of any of these registers is set to Disconnected, the associated SPI slave signal will not be connected to any physical pins.

The PSEL.CSN, PSEL.SCK, PSEL.MOSI, and PSEL.MISO registers and their configurations are only used as long as the SPI slave is enabled, and retained only as long as the device is in System ON mode, see POWER — Power supply on page 48 chapter for more information about power modes. When the peripheral is disabled, the pins will behave as regular GPIOs, and use the configuration in their respective OUT bit field and PIN_CNF[n] register. PSEL.CSN, PSEL.SCK, PSEL.MOSI, and PSEL.MISO must only be configured when the SPI slave is disabled.

To secure correct behavior in the SPI slave, the pins used by the SPI slave must be configured in the GPIO peripheral as described in GPIO configuration before enabling peripheral on page 307 before enabling the SPI slave. This is to secure that the pins used by the SPI slave are driven correctly if the SPI slave itself is temporarily disabled, or if the device temporarily enters System OFF. This configuration must be retained in the GPIO for the selected I/Os as long as the SPI slave is to be recognized by an external SPI master.

The MISO line is set in high impedance as long as the SPI slave is not selected with CSN.

Only one peripheral can be assigned to drive a particular GPIO pin at a time. Failing to do so may result in unpredictable behavior.

| SPI signal | SPI pin | Direction | Output value | Comment |
|---|---|---|---|---|
| CSN | As specified in PSEL.CSN | Input | Not applicable | |
| SCK | As specified in PSEL.SCK | Input | Not applicable | |
| MOSI | As specified in PSEL.MOSI | Input | Not applicable | |
| MISO | As specified in PSEL.MISO | Input | Not applicable | Emulates that the SPI slave is not selected. |

*Table 92: GPIO configuration before enabling peripheral*

## 6.20.5 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40003000 | SPIS | SPIS1 | SPI slave 1 | |
| 0x40004000 | SPIS | SPIS0 | SPI slave 0 | |

*Table 93: Instances*

| Register | Offset | Description | |
|---|---|---|---|
| TASKS_ACQUIRE | 0x024 | Acquire SPI semaphore | |
| TASKS_RELEASE | 0x028 | Release SPI semaphore, enabling the SPI slave to acquire it | |
| EVENTS_END | 0x104 | Granted transaction completed | |
| EVENTS_ENDRX | 0x110 | End of RXD buffer reached | |
| EVENTS_ACQUIRED | 0x128 | Semaphore acquired | |
| SHORTS | 0x200 | Shortcuts between local events and tasks | |
| INTENSET | 0x304 | Enable interrupt | |
| INTENCLR | 0x308 | Disable interrupt | |
| SEMSTAT | 0x400 | Semaphore status register | |
| STATUS | 0x440 | Status from last transaction | |
| ENABLE | 0x500 | Enable SPI slave | |
| PSEL.SCK | 0x508 | Pin select for SCK | |
| PSEL.MISO | 0x50C | Pin select for MISO signal | |
| PSEL.MOSI | 0x510 | Pin select for MOSI signal | |
| PSEL.CSN | 0x514 | Pin select for CSN signal | |
| PSELSCK | 0x508 | Pin select for SCK | Deprecated |
| PSELMISO | 0x50C | Pin select for MISO | Deprecated |
| PSELMOSI | 0x510 | Pin select for MOSI | Deprecated |
| PSELCSN | 0x514 | Pin select for CSN | Deprecated |
| RXDPTR | 0x534 | RXD data pointer | Deprecated |
| MAXRX | 0x538 | Maximum number of bytes in receive buffer | Deprecated |
| AMOUNTRX | 0x53C | Number of bytes received in last granted transaction | Deprecated |
| RXD.PTR | 0x534 | RXD data pointer | |
| RXD.MAXCNT | 0x538 | Maximum number of bytes in receive buffer | |
| RXD.AMOUNT | 0x53C | Number of bytes received in last granted transaction | |
| RXD.LIST | 0x540 | EasyDMA list type | |
| TXDPTR | 0x544 | TXD data pointer | Deprecated |
| MAXTX | 0x548 | Maximum number of bytes in transmit buffer | Deprecated |
| AMOUNTTX | 0x54C | Number of bytes transmitted in last granted transaction | Deprecated |
| TXD.PTR | 0x544 | TXD data pointer | |
| TXD.MAXCNT | 0x548 | Maximum number of bytes in transmit buffer | |
| TXD.AMOUNT | 0x54C | Number of bytes transmitted in last granted transaction | |
| TXD.LIST | 0x550 | EasyDMA list type | |

| Register | Offset | Description |
|---|---|---|
| CONFIG | 0x554 | Configuration register |
| DEF | 0x55C | Default character. Character clocked out in case of an ignored transaction. |
| ORC | 0x5C0 | Over-read character |

*Table 94: Register overview*

### 6.20.5.1 TASKS_ACQUIRE

Address offset: 0x024

Acquire SPI semaphore

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_ACQUIRE | | | Acquire SPI semaphore |
| | | | Trigger | 1 | Trigger task |

### 6.20.5.2 TASKS_RELEASE

Address offset: 0x028

Release SPI semaphore, enabling the SPI slave to acquire it

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_RELEASE | | | Release SPI semaphore, enabling the SPI slave to acquire it |
| | | | Trigger | 1 | Trigger task |

### 6.20.5.3 EVENTS_END

Address offset: 0x104

Granted transaction completed

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | EVENTS_END | | | Granted transaction completed |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.20.5.4 EVENTS_ENDRX

Address offset: 0x110

End of RXD buffer reached

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_ENDRX | | | End of RXD buffer reached |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.20.5.5 EVENTS_ACQUIRED

Address offset: 0x128

Semaphore acquired

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_ACQUIRED | | | Semaphore acquired |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.20.5.6 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | END_ACQUIRE | | | Shortcut between event END and task ACQUIRE |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |

## 6.20.5.7 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | END | | | Write '1' to enable interrupt for event END |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | ENDRX | | | Write '1' to enable interrupt for event ENDRX |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | ACQUIRED | | | Write '1' to enable interrupt for event ACQUIRED |

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | C B A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.20.5.8 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | C B A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | END | | | Write '1' to disable interrupt for event END |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | ENDRX | | | Write '1' to disable interrupt for event ENDRX |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | ACQUIRED | | | Write '1' to disable interrupt for event ACQUIRED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.20.5.9 SEMSTAT

Address offset: 0x400

Semaphore status register

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A |
| **Reset 0x00000001** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | SEMSTAT | | | Semaphore status |
| | | | Free | 0 | Semaphore is free |
| | | | CPU | 1 | Semaphore is assigned to CPU |
| | | | SPIS | 2 | Semaphore is assigned to SPI slave |
| | | | CPUPending | 3 | Semaphore is assigned to SPI but a handover to the CPU is pending |

## 6.20.5.10 STATUS

Address offset: 0x440

Status from last transaction

Individual bits are cleared by writing a '1' to the bits that shall be cleared

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | B A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | OVERREAD | | | TX buffer over-read detected, and prevented |
| | | | NotPresent | 0 | Read: error not present |
| | | | Present | 1 | Read: error present |
| | | | Clear | 1 | Write: clear error on writing '1' |
| B | RW | OVERFLOW | | | RX buffer overflow detected, and prevented |
| | | | NotPresent | 0 | Read: error not present |
| | | | Present | 1 | Read: error present |
| | | | Clear | 1 | Write: clear error on writing '1' |

## 6.20.5.11 ENABLE

Address offset: 0x500

Enable SPI slave

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | ENABLE | | | Enable or disable SPI slave |
| | | | Disabled | 0 | Disable SPI slave |
| | | | Enabled | 2 | Enable SPI slave |

## 6.20.5.12 PSEL.SCK

Address offset: 0x508

Pin select for SCK

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.20.5.13 PSEL.MISO

Address offset: 0x50C

Pin select for MISO signal

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.20.5.14 PSEL.MOSI

Address offset: 0x510

Pin select for MOSI signal

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.20.5.15 PSEL.CSN

Address offset: 0x514

Pin select for CSN signal

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.20.5.16 PSELSCK ( Deprecated )

Address offset: 0x508

Pin select for SCK

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0xFFFFFFFF** | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | PSELSCK | | [0..31] | Pin number configuration for SPI SCK signal |
| | | | Disconnected | 0xFFFFFFFF | Disconnect |

## 6.20.5.17 PSELMISO ( Deprecated )

Address offset: 0x50C

Pin select for MISO

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PSELMISO | | [0..31] | Pin number configuration for SPI MISO signal |
| | | | Disconnected | 0xFFFFFFFF | Disconnect |

## 6.20.5.18 PSELMOSI ( Deprecated )

Address offset: 0x510

Pin select for MOSI

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PSELMOSI | | [0..31] | Pin number configuration for SPI MOSI signal |
| | | | Disconnected | 0xFFFFFFFF | Disconnect |

## 6.20.5.19 PSELCSN ( Deprecated )

Address offset: 0x514

Pin select for CSN

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PSELCSN | | [0..31] | Pin number configuration for SPI CSN signal |
| | | | Disconnected | 0xFFFFFFFF | Disconnect |

## 6.20.5.20 RXDPTR ( Deprecated )

Address offset: 0x534

RXD data pointer

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | RXDPTR | | | RXD data pointer |

**Note:** See the memory chapter for details about which memories are available for EasyDMA.

## 6.20.5.21 MAXRX ( Deprecated )

Address offset: 0x538

Maximum number of bytes in receive buffer

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | MAXRX | | [1..0x3FFF] | Maximum number of bytes in receive buffer |

## 6.20.5.22 AMOUNTRX ( Deprecated )

Address offset: 0x53C

Number of bytes received in last granted transaction

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | AMOUNTRX | | [1..0x3FFF] | Number of bytes received in the last granted transaction |

## 6.20.5.23 RXD.PTR

Address offset: 0x534

RXD data pointer

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | PTR | | | RXD data pointer |

**Note:** See the memory chapter for details about which memories are available for EasyDMA.

## 6.20.5.24 RXD.MAXCNT

Address offset: 0x538

Maximum number of bytes in receive buffer

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | MAXCNT | | [1..0x3FFF] | Maximum number of bytes in receive buffer |

## 6.20.5.25 RXD.AMOUNT

Address offset: 0x53C

Number of bytes received in last granted transaction

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | AMOUNT | | [1..0x3FFF] | Number of bytes received in the last granted transaction |

## 6.20.5.26 RXD.LIST

Address offset: 0x540

EasyDMA list type

| Bit number | 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 |
|---|---|
| ID | A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | LIST | | | List type |
| | | | Disabled | 0 | Disable EasyDMA list |
| | | | ArrayList | 1 | Use array list |

## 6.20.5.27 TXDPTR ( Deprecated )

Address offset: 0x544

TXD data pointer

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | TXDPTR | | | TXD data pointer |

**Note:** See the memory chapter for details about which memories are available for EasyDMA.

## 6.20.5.28 MAXTX ( Deprecated )

Address offset: 0x548

Maximum number of bytes in transmit buffer

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | MAXTX | | [1..0x3FFF] | Maximum number of bytes in transmit buffer |

## 6.20.5.29 AMOUNTTX ( Deprecated )

Address offset: 0x54C

Number of bytes transmitted in last granted transaction

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | AMOUNTTX | | [1..0x3FFF] | Number of bytes transmitted in last granted transaction |

## 6.20.5.30 TXD.PTR

Address offset: 0x544

TXD data pointer

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | PTR | | | TXD data pointer |

**Note:** See the memory chapter for details about which memories are available for EasyDMA.

## 6.20.5.31 TXD.MAXCNT

Address offset: 0x548

Maximum number of bytes in transmit buffer

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | MAXCNT | | [1..0x3FFF] | Maximum number of bytes in transmit buffer |

## 6.20.5.32 TXD.AMOUNT

Address offset: 0x54C

Number of bytes transmitted in last granted transaction

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | AMOUNT | | [1..0x3FFF] | Number of bytes transmitted in last granted transaction |

## 6.20.5.33 TXD.LIST

Address offset: 0x550

EasyDMA list type

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | LIST | | | List type |
| | | | Disabled | 0 | Disable EasyDMA list |
| | | | ArrayList | 1 | Use array list |

## 6.20.5.34 CONFIG

Address offset: 0x554

Configuration register

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | C B A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | ORDER | | | Bit order |
| | | | MsbFirst | 0 | Most significant bit shifted out first |
| | | | LsbFirst | 1 | Least significant bit shifted out first |
| B | RW | CPHA | | | Serial clock (SCK) phase |
| | | | Leading | 0 | Sample on leading edge of clock, shift serial data on trailing edge |
| | | | Trailing | 1 | Sample on trailing edge of clock, shift serial data on leading edge |
| C | RW | CPOL | | | Serial clock (SCK) polarity |
| | | | ActiveHigh | 0 | Active high |
| | | | ActiveLow | 1 | Active low |

## 6.20.5.35 DEF

Address offset: 0x55C

Default character. Character clocked out in case of an ignored transaction.

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | DEF | | | Default character. Character clocked out in case of an ignored transaction. |

## 6.20.5.36 ORC

Address offset: 0x5C0

Over-read character

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | ORC | | | Over-read character. Character clocked out after an over-read of the transmit buffer. |

## 6.20.6 Electrical specification

### 6.20.6.1 SPIS slave interface electrical specifications

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $f_{SPIS}$ | Bit rates for SPIS[26] | | | 8[27] | Mbps |
| $t_{SPIS,START}$ | Time from RELEASE task to receive/transmit (CSN active) | .. | .. | .. | µs |

### 6.20.6.2 Serial Peripheral Interface Slave (SPIS) timing specifications

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{SPIS,CSCKIN}$ | SCK input period | .. | .. | .. | ns |
| $t_{SPIS,RFSCKIN}$ | SCK input rise/fall time | | | 30 | ns |
| $t_{SPIS,WHSCKIN}$ | SCK input high time | 30 | | | ns |
| $t_{SPIS,WLSCKIN}$ | SCK input low time | 30 | | | ns |
| $t_{SPIS,SUCSN}$ | CSN to CLK setup time | .. | .. | .. | ns |
| $t_{SPIS,HCSN}$ | CLK to CSN hold time | 2000 | | | ns |
| $t_{SPIS,ASA}$ | CSN to MISO driven | .. | .. | .. | ns |
| $t_{SPIS,ASO}$ | CSN to MISO valid[a] | | | 1000 | ns |
| $t_{SPIS,DISSO}$ | CSN to MISO disabled[a] | | | 68 | ns |
| $t_{SPIS,CWH}$ | CSN inactive time | 300 | | | ns |
| $t_{SPIS,VSO}$ | CLK edge to MISO valid | | | 19 | ns |
| $t_{SPIS,HSO}$ | MISO hold time after CLK edge | 18[28] | | | ns |
| $t_{SPIS,SUSI}$ | MOSI to CLK edge setup time | 59 | | | ns |
| $t_{SPIS,HSI}$ | CLK edge to MOSI hold time | 20 | | | ns |

[26] High bit rates may require GPIOs to be set as High Drive, see GPIO chapter for more details.

[27] The actual maximum data rate depends on the master's CLK to MISO and MOSI setup and hold timings.

[a] At 25pF load, including GPIO capacitance, see GPIO spec.

[28] This is to ensure compatibility to SPI masters sampling MISO on the same edge as MOSI is output

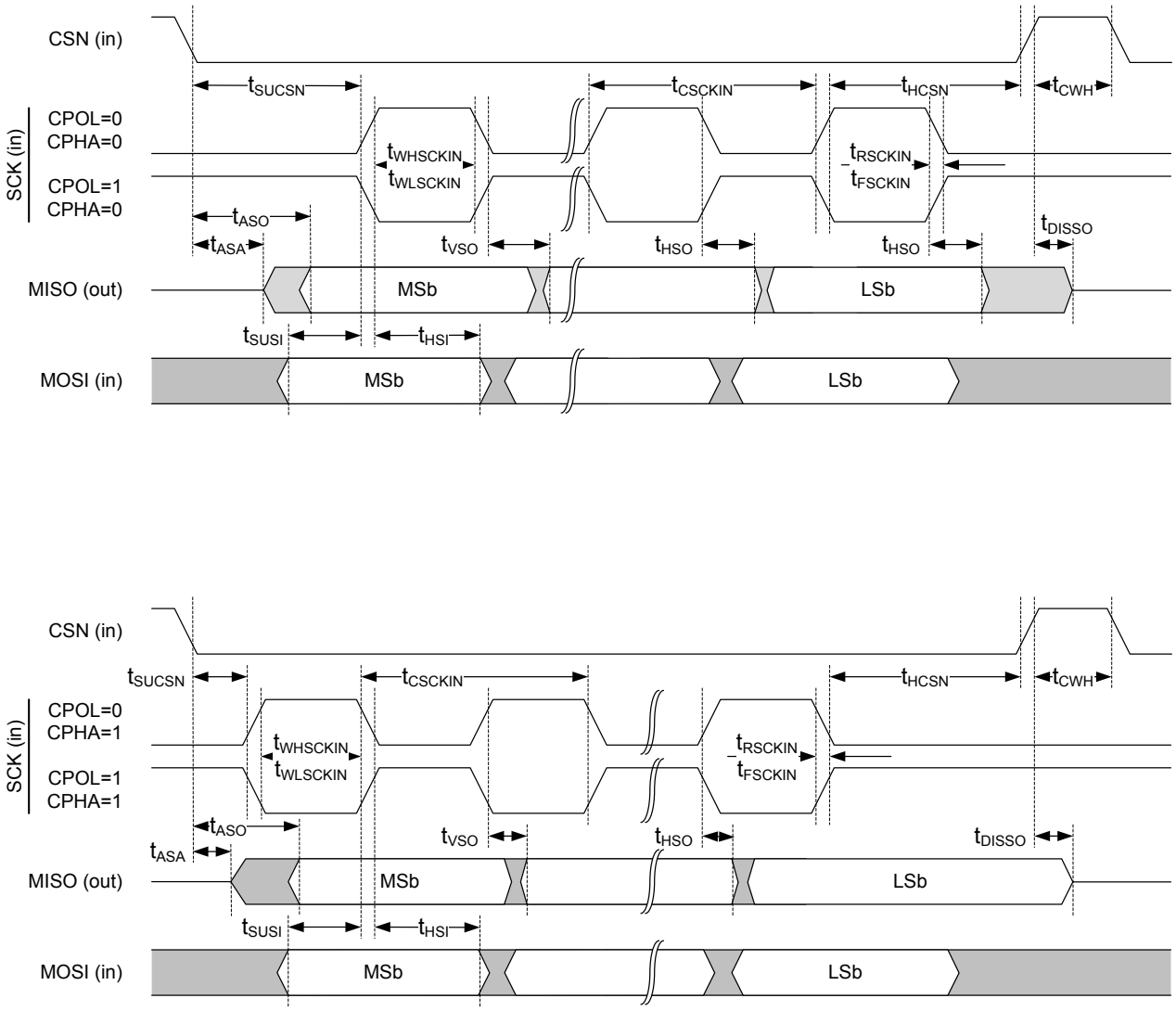



*Figure 120: SPIS timing diagram*

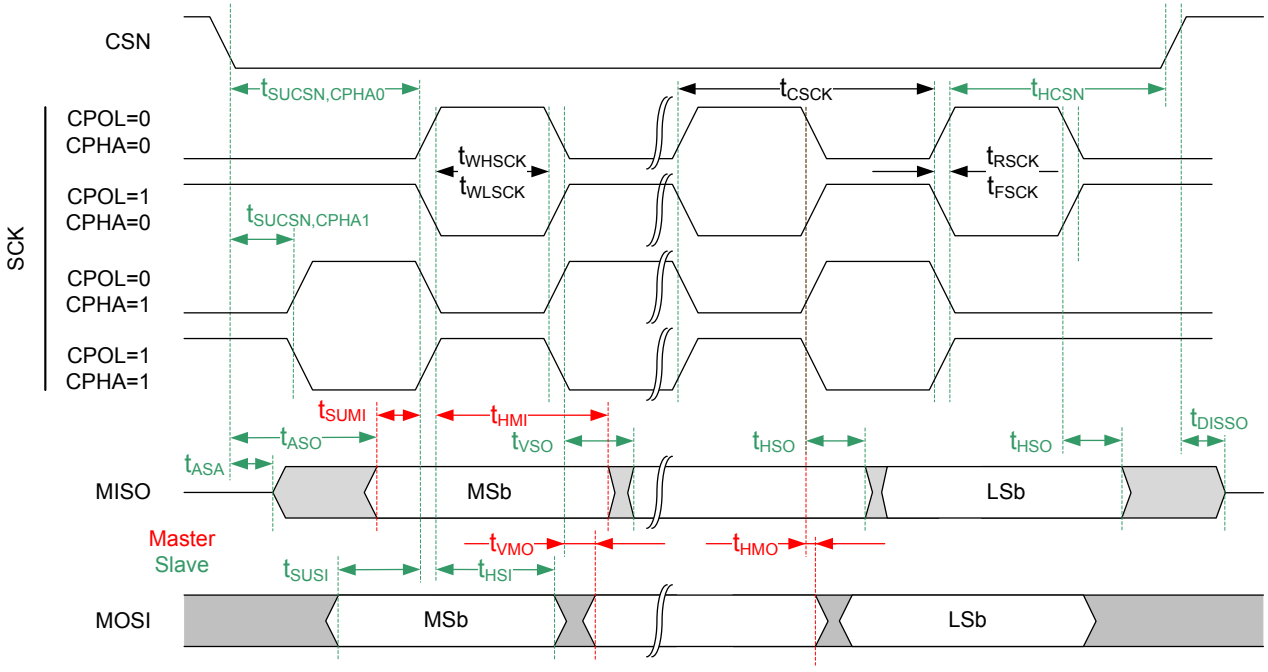



*Figure 121: Common SPIM and SPIS timing diagram*

# 6.21 SWI — Software interrupts

A set of interrupts have been reserved for use as software interrupts.

## 6.21.1 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40014000 | SWI | SWI0 | Software interrupt 0 | |
| 0x40015000 | SWI | SWI1 | Software interrupt 1 | |
| 0x40016000 | SWI | SWI2 | Software interrupt 2 | |
| 0x40017000 | SWI | SWI3 | Software interrupt 3 | |
| 0x40018000 | SWI | SWI4 | Software interrupt 4 | |
| 0x40019000 | SWI | SWI5 | Software interrupt 5 | |

*Table 95: Instances*

# 6.22 TEMP — Temperature sensor

The temperature sensor measures die temperature over the temperature range of the device. Linearity compensation can be implemented if required by the application.

Listed here are the main features for TEMP:

- Temperature range is greater than or equal to operating temperature of the device
- Resolution is 0.25 degrees

TEMP is started by triggering the START task.

When the temperature measurement is completed, a DATARDY event will be generated and the result of the measurement can be read from the TEMP register.

To achieve the measurement accuracy stated in the electrical specification, the crystal oscillator must be selected as the HFCLK source, see CLOCK — Clock control on page 61 for more information.

When the temperature measurement is completed, TEMP analog electronics power down to save power.

TEMP only supports one-shot operation, meaning that every TEMP measurement has to be explicitly started using the START task.

## 6.22.1 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x4000C000 | TEMP | TEMP | Temperature sensor | |

*Table 96: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_START | 0x000 | Start temperature measurement |
| TASKS_STOP | 0x004 | Stop temperature measurement |
| EVENTS_DATARDY | 0x100 | Temperature measurement complete, data ready |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| TEMP | 0x508 | Temperature in °C (0.25° steps) |
| A0 | 0x520 | Slope of 1st piece wise linear function |

| Register | Offset | Description |
|---|---|---|
| A1 | 0x524 | Slope of 2nd piece wise linear function |
| A2 | 0x528 | Slope of 3rd piece wise linear function |
| A3 | 0x52C | Slope of 4th piece wise linear function |
| A4 | 0x530 | Slope of 5th piece wise linear function |
| A5 | 0x534 | Slope of 6th piece wise linear function |
| B0 | 0x540 | y-intercept of 1st piece wise linear function |
| B1 | 0x544 | y-intercept of 2nd piece wise linear function |
| B2 | 0x548 | y-intercept of 3rd piece wise linear function |
| B3 | 0x54C | y-intercept of 4th piece wise linear function |
| B4 | 0x550 | y-intercept of 5th piece wise linear function |
| B5 | 0x554 | y-intercept of 6th piece wise linear function |
| T0 | 0x560 | End point of 1st piece wise linear function |
| T1 | 0x564 | End point of 2nd piece wise linear function |
| T2 | 0x568 | End point of 3rd piece wise linear function |
| T3 | 0x56C | End point of 4th piece wise linear function |
| T4 | 0x570 | End point of 5th piece wise linear function |

*Table 97: Register overview*

### 6.22.1.1 TASKS_START

Address offset: 0x000

Start temperature measurement

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_START | | | Start temperature measurement |
| | | | Trigger | 1 | Trigger task |

### 6.22.1.2 TASKS_STOP

Address offset: 0x004

Stop temperature measurement

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_STOP | | | Stop temperature measurement |
| | | | Trigger | 1 | Trigger task |

### 6.22.1.3 EVENTS_DATARDY

Address offset: 0x100

Temperature measurement complete, data ready

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_DATARDY | | | Temperature measurement complete, data ready |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.22.1.4 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | DATARDY | | | Write '1' to enable interrupt for event DATARDY |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.22.1.5 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | DATARDY | | | Write '1' to disable interrupt for event DATARDY |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.22.1.6 TEMP

Address offset: 0x508

Temperature in °C (0.25° steps)

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | TEMP | | | Temperature in °C (0.25° steps)<br>Result of temperature measurement. Die temperature in °C, 2's complement format, 0.25 °C steps<br>Decision point: DATARDY |

### 6.22.1.7 A0

Address offset: 0x520

Slope of 1st piece wise linear function

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000326** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 1 1 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | A0 | | | Slope of 1st piece wise linear function |

### 6.22.1.8 A1

Address offset: 0x524

Slope of 2nd piece wise linear function

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000348** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 1 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | A1 | | | Slope of 2nd piece wise linear function |

### 6.22.1.9 A2

Address offset: 0x528

Slope of 3rd piece wise linear function

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x000003AA** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 1 0 1 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | A2 | | | Slope of 3rd piece wise linear function |

### 6.22.1.10 A3

Address offset: 0x52C

Slope of 4th piece wise linear function

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x0000040E** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 1 1 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | A3 | | | Slope of 4th piece wise linear function |

### 6.22.1.11 A4

Address offset: 0x530

Slope of 5th piece wise linear function

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x000004BD** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 1 1 1 0 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | A4 | | | Slope of 5th piece wise linear function |

## 6.22.1.12 A5

Address offset: 0x534

Slope of 6th piece wise linear function

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x000005A3** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0 1 0 0 0 1 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | A5 | | | Slope of 6th piece wise linear function |

## 6.22.1.13 B0

Address offset: 0x540

y-intercept of 1st piece wise linear function

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00003FEF** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 0 1 1 1 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | B0 | | | y-intercept of 1st piece wise linear function |

## 6.22.1.14 B1

Address offset: 0x544

y-intercept of 2nd piece wise linear function

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00003FBE** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 0 1 1 1 1 1 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | B1 | | | y-intercept of 2nd piece wise linear function |

## 6.22.1.15 B2

Address offset: 0x548

y-intercept of 3rd piece wise linear function

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00003FBE** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 0 1 1 1 1 1 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | B2 | | | y-intercept of 3rd piece wise linear function |

### 6.22.1.16 B3

Address offset: 0x54C

y-intercept of 4th piece wise linear function

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000012** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | B3 | | | y-intercept of 4th piece wise linear function |

### 6.22.1.17 B4

Address offset: 0x550

y-intercept of 5th piece wise linear function

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x00000124** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | B4 | | | y-intercept of 5th piece wise linear function |

### 6.22.1.18 B5

Address offset: 0x554

y-intercept of 6th piece wise linear function

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A |
| **Reset 0x0000027C** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 1 1 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | B5 | | | y-intercept of 6th piece wise linear function |

### 6.22.1.19 T0

Address offset: 0x560

End point of 1st piece wise linear function

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A |
| **Reset 0x000000E2** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 1 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | T0 | | | End point of 1st piece wise linear function |

### 6.22.1.20 T1

Address offset: 0x564

End point of 2nd piece wise linear function

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | T1 | | | End point of 2nd piece wise linear function |

### 6.22.1.21 T2

Address offset: 0x568

End point of 3rd piece wise linear function

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A |
| **Reset 0x00000019** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | T2 | | | End point of 3rd piece wise linear function |

### 6.22.1.22 T3

Address offset: 0x56C

End point of 4th piece wise linear function

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A |
| **Reset 0x0000003C** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | T3 | | | End point of 4th piece wise linear function |

### 6.22.1.23 T4

Address offset: 0x570

End point of 5th piece wise linear function

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A |
| **Reset 0x00000050** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | T4 | | | End point of 5th piece wise linear function |

## 6.22.2 Electrical specification

### 6.22.2.1 Temperature Sensor Electrical Specification

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{TEMP}$ | Time required for temperature measurement | | 36 | | µs |
| $T_{TEMP,RANGE}$ | Temperature sensor range | -40 | | 85 | °C |
| $T_{TEMP,ACC}$ | Temperature sensor accuracy | -5 | | 5 | °C |
| $T_{TEMP,RES}$ | Temperature sensor resolution | | 0.25 | | °C |
| $T_{TEMP,STB}$ | Sample to sample stability at constant device temperature | | +/-0.25 | | °C |
| $T_{TEMP,OFFST}$ | Sample offset at 25°C | -2.5 | | 2.5 | °C |

# 6.23 TWI — I²C compatible two-wire interface

The TWI master is compatible with I²C operating at 100 kHz and 400 kHz.

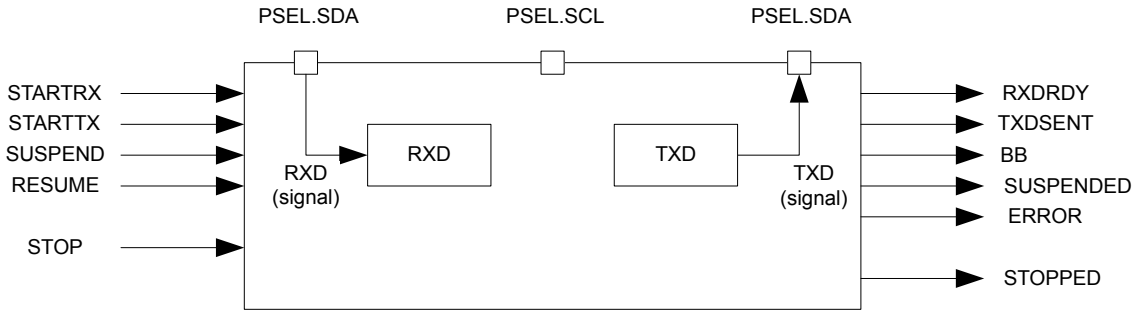



*Figure 122: TWI master's main features*

## 6.23.1 Functional description

This TWI master is not compatible with CBUS. The TWI transmitter and receiver are single buffered.

See, TWI master's main features on page 327.

A TWI setup comprising one master and three slaves is illustrated in A typical TWI setup comprising one master and three slaves on page 327. This TWI master is only able to operate as the only master on the TWI bus.

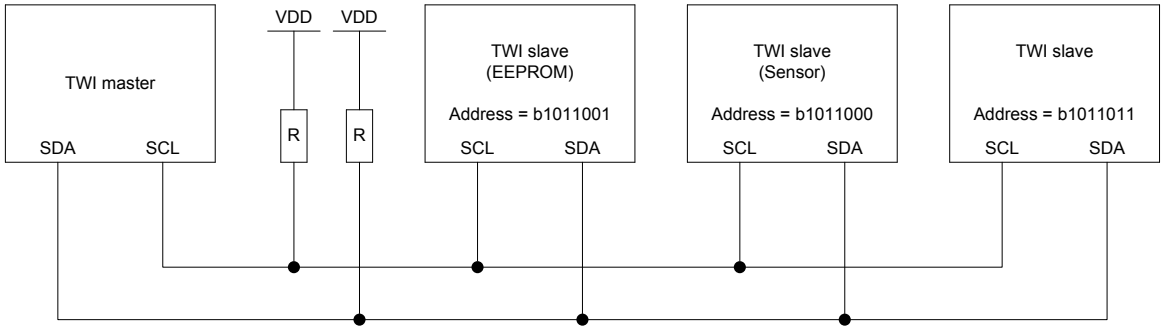



*Figure 123: A typical TWI setup comprising one master and three slaves*

This TWI master supports clock stretching performed by the slaves. The TWI master is started by triggering the STARTTX or STARTRX tasks, and stopped by triggering the STOP task.

If a NACK is clocked in from the slave, the TWI master will generate an ERROR event.

## 6.23.2 Master mode pin configuration

The different signals SCL and SDA associated with the TWI master are mapped to physical pins according to the configuration specified in the PSEL.SCL and PSEL.SDA registers respectively.

If the CONNECT field of a PSEL.xxx register is set to Disconnected, the associated TWI signal is not connected to any physical pin. The PSEL.SCL and PSEL.SDA registers and their configurations are only used

as long as the TWI master is enabled, and retained only as long as the device is in ON mode. PSEL.SCL and PSEL.SDA must only be configured when the TWI is disabled.

To secure correct signal levels on the pins used by the TWI master when the system is in OFF mode, and when the TWI master is disabled, these pins must be configured in the GPIO peripheral as described in GPIO configuration on page 328.

Only one peripheral can be assigned to drive a particular GPIO pin at a time, failing to do so may result in unpredictable behavior.

| TWI master signal | TWI master pin | Direction | Drive strength | Output value |
|---|---|---|---|---|
| SCL | As specified in PSEL.SCL | Input | S0D1 | Not applicable |
| SDA | As specified in PSEL.SDA | Input | S0D1 | Not applicable |

*Table 98: GPIO configuration*

## 6.23.3 Shared resources

The TWI shares registers and other resources with other peripherals that have the same ID as the TWI.

Therefore, you must disable all peripherals that have the same ID as the TWI before the TWI can be configured and used. Disabling a peripheral that has the same ID as the TWI will not reset any of the registers that are shared with the TWI. It is therefore important to configure all relevant TWI registers explicitly to secure that it operates correctly.

The Instantiation table in Instantiation on page 18 shows which peripherals have the same ID as the TWI.

## 6.23.4 Master write sequence

A TWI master write sequence is started by triggering the STARTTX task. After the STARTTX task has been triggered, the TWI master will generate a start condition on the TWI bus, followed by clocking out the address and the READ/WRITE bit set to 0 (WRITE=0, READ=1).

The address must match the address of the slave device that the master wants to write to. The READ/WRITE bit is followed by an ACK/NACK bit (ACK=0 or NACK=1) generated by the slave.

After receiving the ACK bit, the TWI master will clock out the data bytes that are written to the TXD register. Each byte clocked out from the master will be followed by an ACK/NACK bit clocked in from the slave. A TXDSENT event will be generated each time the TWI master has clocked out a TXD byte, and the associated ACK/NACK bit has been clocked in from the slave.

The TWI master transmitter is single buffered, and a second byte can only be written to the TXD register after the previous byte has been clocked out and the ACK/NACK bit clocked in, that is, after the TXDSENT event has been generated.

If the CPU is prevented from writing to TXD when the TWI master is ready to clock out a byte, the TWI master will stretch the clock until the CPU has written a byte to the TXD register.

A typical TWI master write sequence is illustrated in The TWI master writing data to a slave on page 329. Occurrence 3 in the figure illustrates delayed processing of the TXDSENT event associated with TXD byte 1. In this scenario the TWI master will stretch the clock to prevent writing erroneous data to the slave.

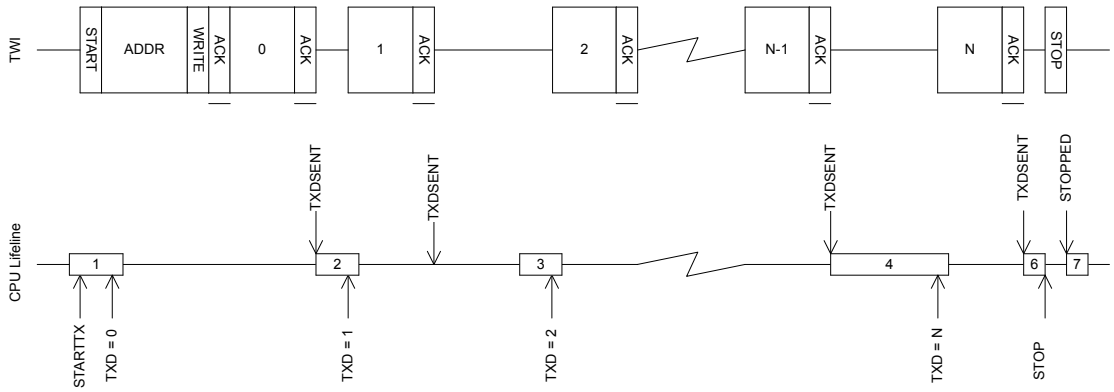



*Figure 124: The TWI master writing data to a slave*

The TWI master write sequence is stopped when the STOP task is triggered whereupon the TWI master will generate a stop condition on the TWI bus.

## 6.23.5 Master read sequence

A TWI master read sequence is started by triggering the STARTRX task. After the STARTRX task has been triggered the TWI master will generate a start condition on the TWI bus, followed by clocking out the address and the READ/WRITE bit set to 1 (WRITE = 0, READ = 1).

The address must match the address of the slave device that the master wants to read from. The READ/WRITE bit is followed by an ACK/NACK bit (ACK=0 or NACK = 1) generated by the slave.

After having sent the ACK bit the TWI slave will send data to the master using the clock generated by the master.

The TWI master will generate a RXDRDY event every time a new byte is received in the RXD register.

After receiving a byte, the TWI master will delay sending the ACK/NACK bit by stretching the clock until the CPU has extracted the received byte, that is, by reading the RXD register.

The TWI master read sequence is stopped by triggering the STOP task. This task must be triggered before the last byte is extracted from RXD to ensure that the TWI master sends a NACK back to the slave before generating the stop condition.

A typical TWI master read sequence is illustrated in The TWI master reading data from a slave on page 330. Occurrence 3 in this figure illustrates delayed processing of the RXDRDY event associated with RXD byte B. In this scenario the TWI master will stretch the clock to prevent the slave from overwriting the contents of the RXD register.

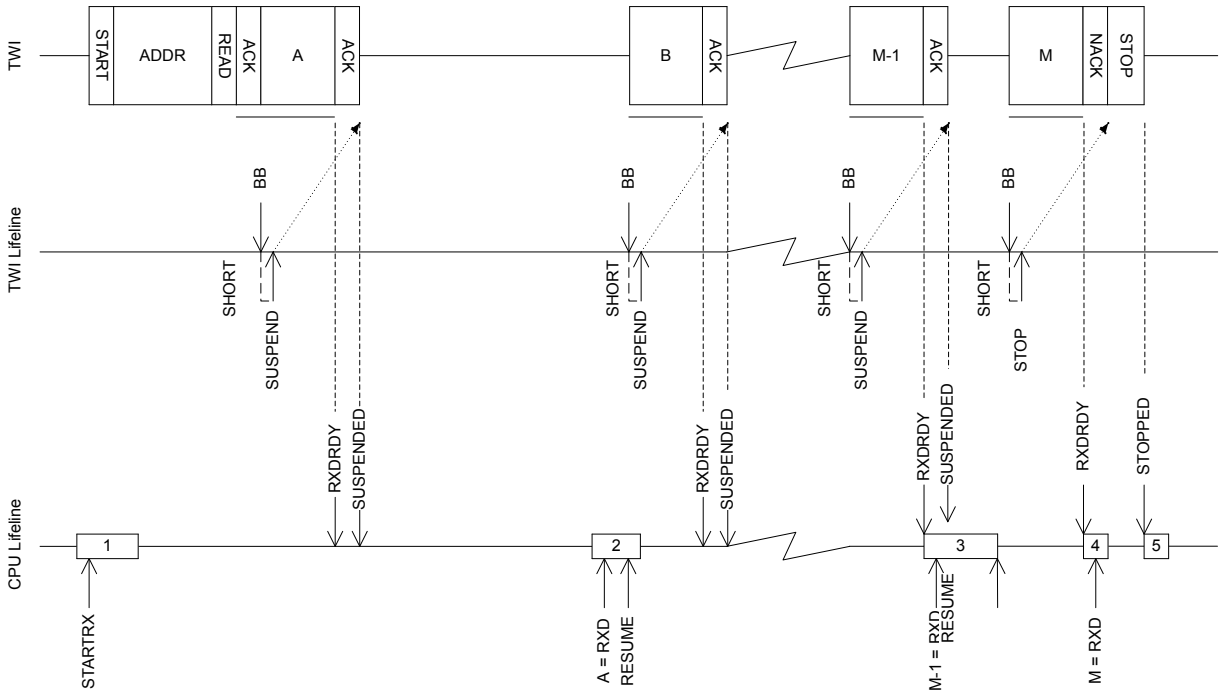



*Figure 125: The TWI master reading data from a slave*

## 6.23.6 Master repeated start sequence

A typical repeated start sequence is one in which the TWI master writes one byte to the slave followed by reading M bytes from the slave. Any combination and number of transmit and receive sequences can be combined in this fashion. Only one shortcut to STOP can be enabled at any given time.

The figure below illustrates a repeated start sequence where the TWI master writes one byte, followed by reading M bytes from the slave without performing a stop in-between.

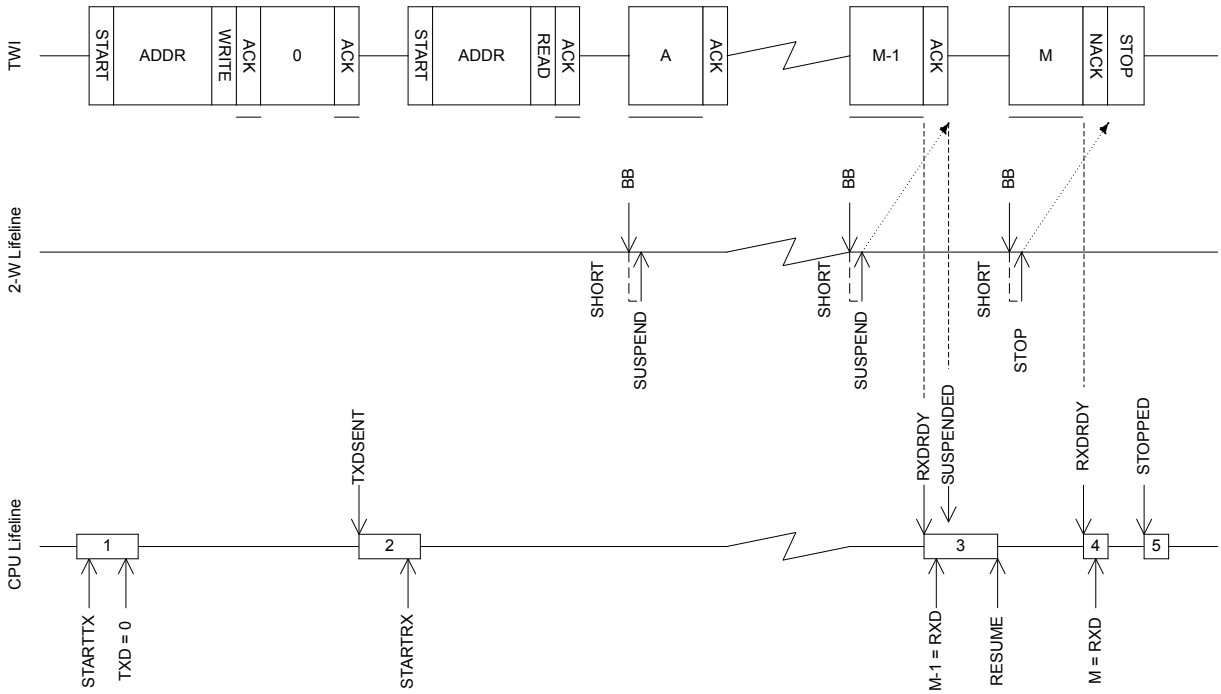



*Figure 126: A repeated start sequence, where the TWI master writes one byte, followed by reading M bytes from the slave without performing a stop in-between*

To generate a repeated start after a read sequence, a second start task must be triggered instead of the STOP task, that is, STARTRX or STARTTX. This start task must be triggered before the last byte is extracted from RXD to ensure that the TWI master sends a NACK back to the slave before generating the repeated start condition.

## 6.23.7 Low power

When putting the system in low power and the peripheral is not needed, lowest possible power consumption is achieved by stopping, and then disabling the peripheral.

The STOP task may not be always needed (the peripheral might already be stopped), but if it is sent, software shall wait until the STOPPED event was received as a response before disabling the peripheral through the ENABLE register.

## 6.23.8 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40003000 | TWI | TWI0 | Two-wire interface master | Deprecated |

*Table 99: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_STARTRX | 0x000 | Start TWI receive sequence |
| TASKS_STARTTX | 0x008 | Start TWI transmit sequence |
| TASKS_STOP | 0x014 | Stop TWI transaction |
| TASKS_SUSPEND | 0x01C | Suspend TWI transaction |
| TASKS_RESUME | 0x020 | Resume TWI transaction |
| EVENTS_STOPPED | 0x104 | TWI stopped |
| EVENTS_RXDREADY | 0x108 | TWI RXD byte received |
| EVENTS_TXDSENT | 0x11C | TWI TXD byte sent |
| EVENTS_ERROR | 0x124 | TWI error |
| EVENTS_BB | 0x138 | TWI byte boundary, generated before each byte that is sent or received |
| EVENTS_SUSPENDED | 0x148 | TWI entered the suspended state |
| SHORTS | 0x200 | Shortcuts between local events and tasks |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| ERRORSRC | 0x4C4 | Error source |
| ENABLE | 0x500 | Enable TWI |
| PSEL.SCL | 0x508 | Pin select for SCL |
| PSEL.SDA | 0x50C | Pin select for SDA |
| RXD | 0x518 | RXD register |
| TXD | 0x51C | TXD register |
| FREQUENCY | 0x524 | TWI frequency. Accuracy depends on the HFCLK source selected. |
| ADDRESS | 0x588 | Address used in the TWI transfer |

*Table 100: Register overview*

### 6.23.8.1 TASKS_STARTRX

Address offset: 0x000

Start TWI receive sequence

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STARTRX | | | Start TWI receive sequence |
| | | | Trigger | 1 | Trigger task |

## 6.23.8.2 TASKS_STARTTX

Address offset: 0x008

Start TWI transmit sequence

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STARTTX | | | Start TWI transmit sequence |
| | | | Trigger | 1 | Trigger task |

## 6.23.8.3 TASKS_STOP

Address offset: 0x014

Stop TWI transaction

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STOP | | | Stop TWI transaction |
| | | | Trigger | 1 | Trigger task |

## 6.23.8.4 TASKS_SUSPEND

Address offset: 0x01C

Suspend TWI transaction

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_SUSPEND | | | Suspend TWI transaction |
| | | | Trigger | 1 | Trigger task |

## 6.23.8.5 TASKS_RESUME

Address offset: 0x020

Resume TWI transaction

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_RESUME | | | Resume TWI transaction |
| | | | Trigger | 1 | Trigger task |

## 6.23.8.6 EVENTS_STOPPED

Address offset: 0x104

TWI stopped

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_STOPPED | | | TWI stopped |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.23.8.7 EVENTS_RXDREADY

Address offset: 0x108

TWI RXD byte received

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_RXDREADY | | | TWI RXD byte received |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.23.8.8 EVENTS_TXDSENT

Address offset: 0x11C

TWI TXD byte sent

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_TXDSENT | | | TWI TXD byte sent |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.23.8.9 EVENTS_ERROR

Address offset: 0x124

TWI error

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | EVENTS_ERROR | | | TWI error |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.23.8.10 EVENTS_BB

Address offset: 0x138

TWI byte boundary, generated before each byte that is sent or received

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | EVENTS_BB | | | TWI byte boundary, generated before each byte that is sent or received |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.23.8.11 EVENTS_SUSPENDED

Address offset: 0x148

TWI entered the suspended state

Generated just after ACK bit has been transferred in a read transaction, and only if SUSPEND has been requested earlier.

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | EVENTS_SUSPENDED | | | TWI entered the suspended state<br>Generated just after ACK bit has been transferred in a read transaction, and only if SUSPEND has been requested earlier. |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.23.8.12 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | BB_SUSPEND | | | Shortcut between event BB and task SUSPEND |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| B | RW | BB_STOP | | | Shortcut between event BB and task STOP |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |

## 6.23.8.13 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | STOPPED | | | Write '1' to enable interrupt for event STOPPED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | RXDREADY | | | Write '1' to enable interrupt for event RXDREADY |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | TXDSENT | | | Write '1' to enable interrupt for event TXDSENT |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | ERROR | | | Write '1' to enable interrupt for event ERROR |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E | RW | BB | | | Write '1' to enable interrupt for event BB |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| F | RW | SUSPENDED | | | Write '1' to enable interrupt for event SUSPENDED<br>Generated just after ACK bit has been transferred in a read transaction, and only if SUSPEND has been requested earlier. |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.23.8.14 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | STOPPED | | | Write '1' to disable interrupt for event STOPPED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | RXDREADY | | | Write '1' to disable interrupt for event RXDREADY |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | TXDSENT | | | Write '1' to disable interrupt for event TXDSENT |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | ERROR | | | Write '1' to disable interrupt for event ERROR |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E | RW | BB | | | Write '1' to disable interrupt for event BB |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| F | RW | SUSPENDED | | | Write '1' to disable interrupt for event SUSPENDED<br>Generated just after ACK bit has been transferred in a read transaction, and only if SUSPEND has been requested earlier. |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.23.8.15 ERRORSRC

Address offset: 0x4C4

Error source

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | OVERRUN | | | Overrun error<br>A new byte was received before previous byte got read by software from the RXD register. (Previous data is lost) |
| | | | NotPresent | 0 | Read: no overrun occured |
| | | | Present | 1 | Read: overrun occured |
| B | RW | ANACK | | | NACK received after sending the address (write '1' to clear) |
| | | | NotPresent | 0 | Read: error not present |
| | | | Present | 1 | Read: error present |
| C | RW | DNACK | | | NACK received after sending a data byte (write '1' to clear) |
| | | | NotPresent | 0 | Read: error not present |
| | | | Present | 1 | Read: error present |

## 6.23.8.16 ENABLE

Address offset: 0x500

Enable TWI

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ENABLE | | | Enable or disable TWI |
| | | | Disabled | 0 | Disable TWI |
| | | | Enabled | 5 | Enable TWI |

## 6.23.8.17 PSEL.SCL

Address offset: 0x508

Pin select for SCL

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.23.8.18 PSEL.SDA

Address offset: 0x50C

Pin select for SDA

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.23.8.19 RXD

Address offset: 0x518

RXD register

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | RXD | | | RXD register |

### 6.23.8.20 TXD

Address offset: 0x51C

TXD register

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | TXD | | | TXD register |

### 6.23.8.21 FREQUENCY

Address offset: 0x524

TWI frequency. Accuracy depends on the HFCLK source selected.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x04000000** | | | | **0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | FREQUENCY | | | TWI master clock frequency |
| | | | K100 | 0x01980000 | 100 kbps |
| | | | K250 | 0x04000000 | 250 kbps |
| | | | K400 | 0x06680000 | 400 kbps (actual rate 410.256 kbps) |

### 6.23.8.22 ADDRESS

Address offset: 0x588

Address used in the TWI transfer

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ADDRESS | | | Address used in the TWI transfer |

## 6.23.9 Electrical specification

### 6.23.9.1 TWI interface electrical specifications

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $f_{TWI,SCL}$ | Bit rates for TWI[29] | 100 | | 400 | kbps |
| $t_{TWI,START}$ | Time from STARTRX/STARTTX task to transmission started | | 1.5 | | µs |

[29] High bit rates or stronger pull-ups may require GPIOs to be set as High Drive, see GPIO chapter for more details.

### 6.23.9.2 Two Wire Interface (TWI) timing specifications

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{TWI,SU\_DAT}$ | Data setup time before positive edge on SCL – all modes | 300 | | | ns |
| $t_{TWI,HD\_DAT}$ | Data hold time after negative edge on SCL – all modes | 500 | | | ns |
| $t_{TWI,HD\_STA,100kbps}$ | TWI master hold time for START and repeated START condition, 100 kbps | 10000 | | | ns |
| $t_{TWI,HD\_STA,250kbps}$ | TWI master hold time for START and repeated START condition, 250kbps | 4000 | | | ns |
| $t_{TWI,HD\_STA,400kbps}$ | TWI master hold time for START and repeated START condition, 400 kbps | 2500 | | | ns |
| $t_{TWI,SU\_STO,100kbps}$ | TWI master setup time from SCL high to STOP condition, 100 kbps | 5000 | | | ns |
| $t_{TWI,SU\_STO,250kbps}$ | TWI master setup time from SCL high to STOP condition, 250 kbps | 2000 | | | ns |
| $t_{TWI,SU\_STO,400kbps}$ | TWI master setup time from SCL high to STOP condition, 400 kbps | 1250 | | | ns |
| $t_{TWI,BUF,100kbps}$ | TWI master bus free time between STOP and START conditions, 100 kbps | 5800 | | | ns |
| $t_{TWI,BUF,250kbps}$ | TWI master bus free time between STOP and START conditions, 250 kbps | 2700 | | | ns |
| $t_{TWI,BUF,400kbps}$ | TWI master bus free time between STOP and START conditions, 400 kbps | 2100 | | | ns |

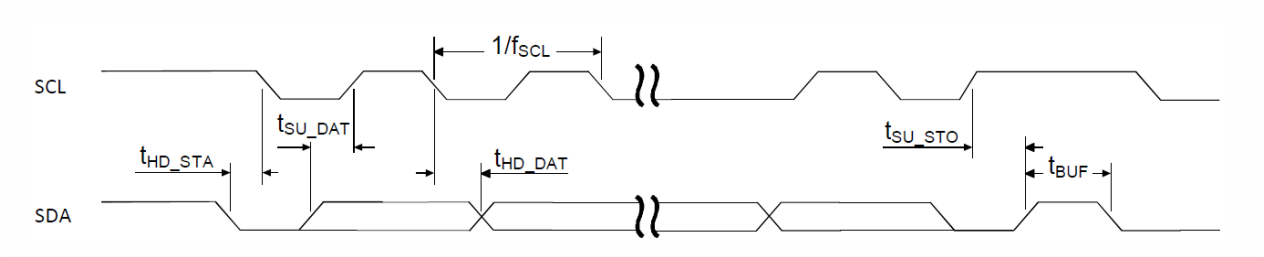



*Figure 127: TWI timing diagram, 1 byte transaction*

# 6.24 TIMER — Timer/counter

The TIMER can operate in two modes: timer and counter.

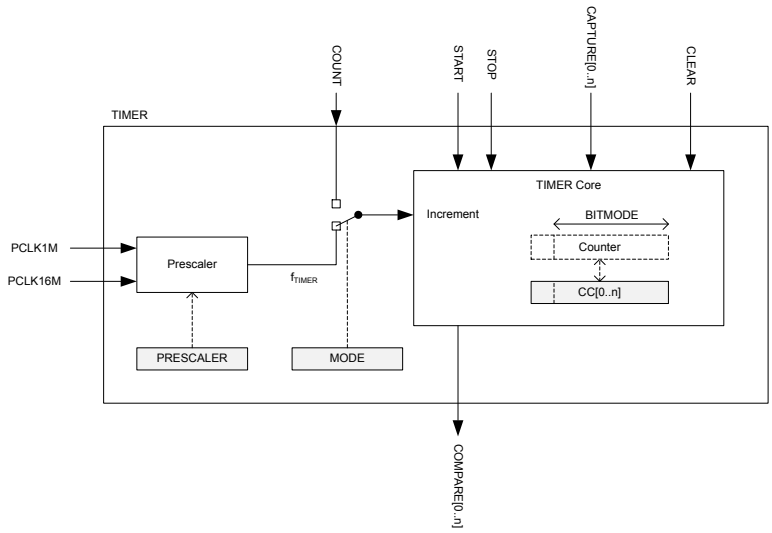



*Figure 128: Block schematic for timer/counter*

The timer/counter runs on the high-frequency clock source (HFCLK) and includes a four-bit (1/2X) prescaler that can divide the timer input clock from the HFCLK controller. Clock source selection between PCLK16M and PCLK1M is automatic according to TIMER base frequency set by the prescaler. The TIMER base frequency is always given as 16 MHz divided by the prescaler value.

The PPI system allows a TIMER event to trigger a task of any other system peripheral of the device. The PPI system also enables the TIMER task/event features to generate periodic output and PWM signals to any GPIO. The number of input/outputs used at the same time is limited by the number of GPIOTE channels.

The TIMER can operate in two modes, Timer mode and Counter mode. In both modes, the TIMER is started by triggering the START task, and stopped by triggering the STOP task. After the timer is stopped the timer can resume timing/counting by triggering the START task again. When timing/counting is resumed, the timer will continue from the value it had prior to being stopped.

In Timer mode, the TIMER's internal Counter register is incremented by one for every tick of the timer frequency $f_{TIMER}$ as illustrated in Block schematic for timer/counter on page 340. The timer frequency is derived from PCLK16M as shown below, using the values specified in the PRESCALER register:

$$f_{TIMER} = 16\ MHz\ /\ (2^{PRESCALER})$$

When $f_{TIMER}$ <= 1 MHz the TIMER will use PCLK1M instead of PCLK16M for reduced power consumption.

In counter mode, the TIMER's internal Counter register is incremented by one each time the COUNT task is triggered, that is, the timer frequency and the prescaler are not utilized in counter mode. Similarly, the COUNT task has no effect in Timer mode.

The TIMER's maximum value is configured by changing the bit-width of the timer in the BITMODE on page 345 register.

PRESCALER on page 345 and the BITMODE on page 345 must only be updated when the timer is stopped. If these registers are updated while the TIMER is started then this may result in unpredictable behavior.

When the timer is incremented beyond its maximum value the Counter register will overflow and the TIMER will automatically start over from zero.

The Counter register can be cleared, that is, its internal value set to zero explicitly, by triggering the CLEAR task.

The TIMER implements multiple capture/compare registers.

Independent of prescaler setting the accuracy of the TIMER is equivalent to one tick of the timer frequency $f_{TIMER}$ as illustrated in Block schematic for timer/counter on page 340.

## 6.24.1 Capture

The TIMER implements one capture task for every available capture/compare register.

Every time the CAPTURE[n] task is triggered, the Counter value is copied to the CC[n] register.

## 6.24.2 Compare

The TIMER implements one COMPARE event for every available capture/compare register.

A COMPARE event is generated when the Counter is incremented and then becomes equal to the value specified in one of the capture compare registers. When the Counter value becomes equal to the value specified in a capture compare register CC[n], the corresponding compare event COMPARE[n] is generated.

BITMODE on page 345 specifies how many bits of the Counter register and the capture/compare register that are used when the comparison is performed. Other bits will be ignored.

## 6.24.3 Task delays

After the TIMER is started, the CLEAR task, COUNT task and the STOP task will guarantee to take effect within one clock cycle of the PCLK16M.

## 6.24.4 Task priority

If the START task and the STOP task are triggered at the same time, that is, within the same period of PCLK16M, the STOP task will be prioritized.

## 6.24.5 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40008000 | TIMER | TIMER0 | Timer 0 | This timer instance has 4 CC registers (CC[0..3]) |
| 0x40009000 | TIMER | TIMER1 | Timer 1 | This timer instance has 4 CC registers (CC[0..3]) |
| 0x4000A000 | TIMER | TIMER2 | Timer 2 | This timer instance has 4 CC registers (CC[0..3]) |

*Table 101: Instances*

| Register | Offset | Description | |
|---|---|---|---|
| TASKS_START | 0x000 | Start Timer | |
| TASKS_STOP | 0x004 | Stop Timer | |
| TASKS_COUNT | 0x008 | Increment Timer (Counter mode only) | |
| TASKS_CLEAR | 0x00C | Clear time | |
| TASKS_SHUTDOWN | 0x010 | Shut down timer | Deprecated |
| TASKS_CAPTURE[0] | 0x040 | Capture Timer value to CC[0] register | |
| TASKS_CAPTURE[1] | 0x044 | Capture Timer value to CC[1] register | |
| TASKS_CAPTURE[2] | 0x048 | Capture Timer value to CC[2] register | |

| Register | Offset | Description |
|---|---|---|
| TASKS_CAPTURE[3] | 0x04C | Capture Timer value to CC[3] register |
| TASKS_CAPTURE[4] | 0x050 | Capture Timer value to CC[4] register |
| TASKS_CAPTURE[5] | 0x054 | Capture Timer value to CC[5] register |
| EVENTS_COMPARE[0] | 0x140 | Compare event on CC[0] match |
| EVENTS_COMPARE[1] | 0x144 | Compare event on CC[1] match |
| EVENTS_COMPARE[2] | 0x148 | Compare event on CC[2] match |
| EVENTS_COMPARE[3] | 0x14C | Compare event on CC[3] match |
| EVENTS_COMPARE[4] | 0x150 | Compare event on CC[4] match |
| EVENTS_COMPARE[5] | 0x154 | Compare event on CC[5] match |
| SHORTS | 0x200 | Shortcuts between local events and tasks |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| MODE | 0x504 | Timer mode selection |
| BITMODE | 0x508 | Configure the number of bits used by the TIMER |
| PRESCALER | 0x510 | Timer prescaler register |
| CC[0] | 0x540 | Capture/Compare register 0 |
| CC[1] | 0x544 | Capture/Compare register 1 |
| CC[2] | 0x548 | Capture/Compare register 2 |
| CC[3] | 0x54C | Capture/Compare register 3 |
| CC[4] | 0x550 | Capture/Compare register 4 |
| CC[5] | 0x554 | Capture/Compare register 5 |

*Table 102: Register overview*

## 6.24.5.1 TASKS_START

Address offset: 0x000

Start Timer

| Bit number | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | A |
| **Reset 0x00000000** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_START | | | Start Timer |
| | | | Trigger | 1 | Trigger task |

## 6.24.5.2 TASKS_STOP

Address offset: 0x004

Stop Timer

| Bit number | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | A |
| **Reset 0x00000000** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_STOP | | | Stop Timer |
| | | | Trigger | 1 | Trigger task |

## 6.24.5.3 TASKS_COUNT

Address offset: 0x008

Increment Timer (Counter mode only)

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_COUNT | | | Increment Timer (Counter mode only) |
| | | | Trigger | 1 | Trigger task |

### 6.24.5.4 TASKS_CLEAR

Address offset: 0x00C

Clear time

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_CLEAR | | | Clear time |
| | | | Trigger | 1 | Trigger task |

### 6.24.5.5 TASKS_SHUTDOWN ( Deprecated )

Address offset: 0x010

Shut down timer

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description | |
|---|---|---|---|---|---|---|
| A | W | TASKS_SHUTDOWN | | | Shut down timer | Deprecated |
| | | | Trigger | 1 | Trigger task | |

### 6.24.5.6 TASKS_CAPTURE[n] (n=0..5)

Address offset: 0x040 + (n × 0x4)

Capture Timer value to CC[n] register

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_CAPTURE | | | Capture Timer value to CC[n] register |
| | | | Trigger | 1 | Trigger task |

### 6.24.5.7 EVENTS_COMPARE[n] (n=0..5)

Address offset: 0x140 + (n × 0x4)

Compare event on CC[n] match

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_COMPARE | | | Compare event on CC[n] match |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.24.5.8 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | L K J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-F | RW | COMPARE[i]_CLEAR (i=0..5) | | | Shortcut between event COMPARE[i] and task CLEAR |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| G-L | RW | COMPARE[i]_STOP (i=0..5) | | | Shortcut between event COMPARE[i] and task STOP |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |

## 6.24.5.9 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-F | RW | COMPARE[i] (i=0..5) | | | Write '1' to enable interrupt for event COMPARE[i] |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.24.5.10 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A-F | RW | COMPARE[i] (i=0..5) | | | Write '1' to disable interrupt for event COMPARE[i] |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.24.5.11 MODE

Address offset: 0x504

Timer mode selection

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | MODE | | | Timer mode |
| | | | Timer | 0 | Select Timer mode |
| | | | Counter | 1 | Select Counter mode |
| | | | LowPowerCounter | 2 | Select Low Power Counter mode |

## 6.24.5.12 BITMODE

Address offset: 0x508

Configure the number of bits used by the TIMER

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | BITMODE | | | Timer bit width |
| | | | 16Bit | 0 | 16 bit timer bit width |
| | | | 08Bit | 1 | 8 bit timer bit width |
| | | | 24Bit | 2 | 24 bit timer bit width |
| | | | 32Bit | 3 | 32 bit timer bit width |

## 6.24.5.13 PRESCALER

Address offset: 0x510

Timer prescaler register

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A |
| **Reset 0x00000004** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PRESCALER | | [0..9] | Prescaler value |

## 6.24.5.14 CC[n] (n=0..5)

Address offset: 0x540 + (n × 0x4)

Capture/Compare register n

| Bit number | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A |
| **Reset 0x00000000** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | CC | | | Capture/Compare value<br>Only the number of bits indicated by BITMODE will be used by the TIMER. |

# 6.25 TWIM — I$^2$C compatible two-wire interface master with EasyDMA

TWI master with EasyDMA (TWIM) is a two-wire half-duplex master which can communicate with multiple slave devices connected to the same bus

Listed here are the main features for TWIM:

- I$^2$C compatible
- Supported baud rates: 100, 250, 400 kbps
- Support for clock stretching (non I$^2$C compliant)
- EasyDMA

The two-wire interface can communicate with a bi-directional wired-AND bus with two lines (SCL, SDA). The protocol makes it possible to interconnect up to 127 individually addressable devices. TWIM is not compatible with CBUS.

The GPIOs used for each two-wire interface line can be chosen from any GPIO on the device and are independently configurable. This enables great flexibility in device pinout and efficient use of board space and signal routing.

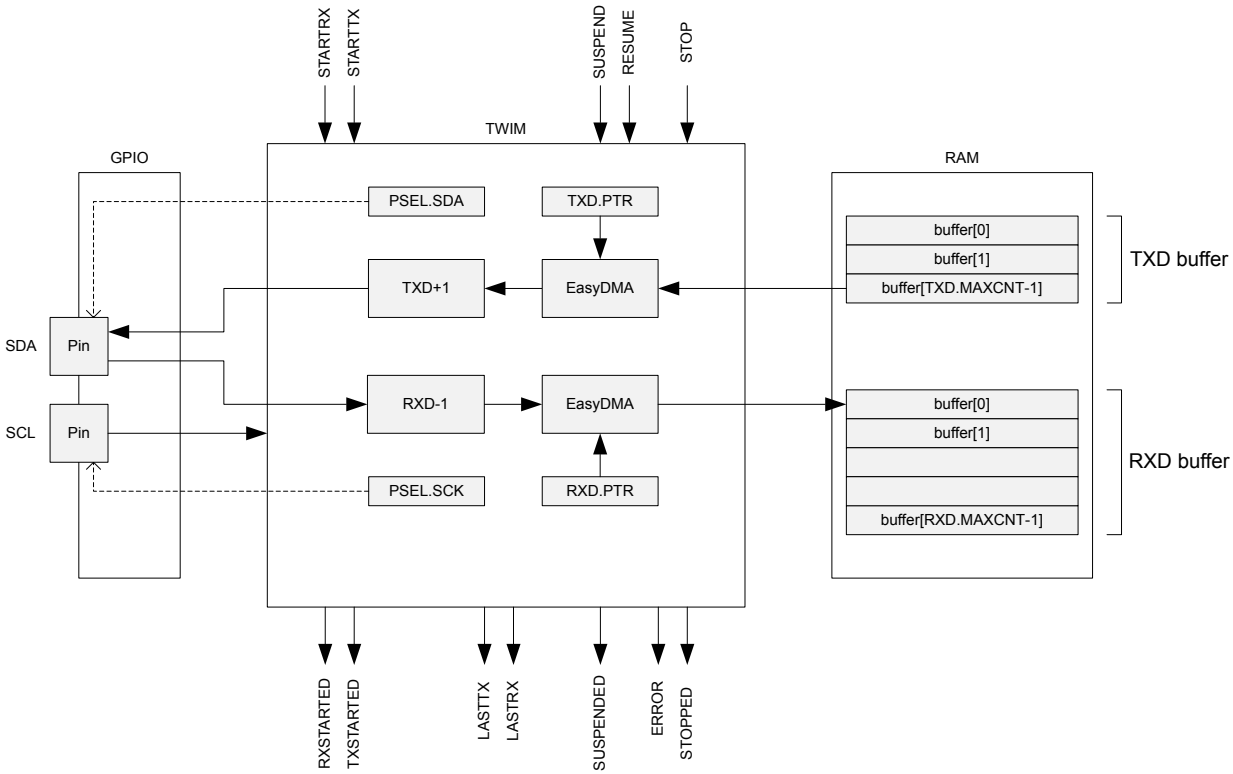



*Figure 129: TWI master with EasyDMA*

A typical TWI setup consists of one master and one or more slaves. For an example, see A typical TWI setup comprising one master and three slaves on page 347. This TWIM is only able to operate as a single master on the TWI bus. Multi-master bus configuration is not supported.

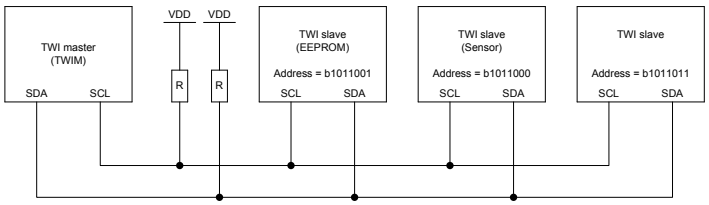



*Figure 130: A typical TWI setup comprising one master and three slaves*

This TWI master supports clock stretching performed by the slaves. Note that the SCK pulse following a stretched clock cycle may be shorter than specified by the I2C specification.

The TWI master is started by triggering the STARTTX or STARTRX tasks, and stopped by triggering the STOP task. The TWI master will generate a STOPPED event when it has stopped following a STOP task. The TWI master cannot get stopped while it is suspended, so the STOP task has to be issued after the TWI master has been resumed.

After the TWI master is started, the STARTTX task or the STARTRX task should not be triggered again before the TWI master has stopped, i.e. following a LASTRX, LASTTX or STOPPED event.

If a NACK is clocked in from the slave, the TWI master will generate an ERROR event.

## 6.25.1 EasyDMA

The TWIM implements EasyDMA for accessing RAM without CPU involvement.

The TWIM peripheral implements the following EasyDMA channels:

| Channel | Type | Register Cluster |
|---|---|---|
| TXD | READER | TXD |
| RXD | WRITER | RXD |

*Table 103: TWIM EasyDMA Channels*

For detailed information regarding the use of EasyDMA, see EasyDMA on page 35.

The .PTR and .MAXCNT registers are double-buffered. They can be updated and prepared for the next RX/TX transmission immediately after having received the RXSTARTED/TXSTARTED event.

The STOPPED event indicates that EasyDMA has finished accessing the buffer in RAM.

## 6.25.2 Master write sequence

A TWI master write sequence is started by triggering the STARTTX task. After the STARTTX task has been triggered, the TWI master will generate a start condition on the TWI bus, followed by clocking out the address and the READ/WRITE bit set to 0 (WRITE=0, READ=1).

The address must match the address of the slave device that the master wants to write to. The READ/WRITE bit is followed by an ACK/NACK bit (ACK=0 or NACK=1) generated by the slave.

After receiving the ACK bit, the TWI master will clock out the data bytes found in the transmit buffer located in RAM at the address specified in the TXD.PTR register. Each byte clocked out from the master will be followed by an ACK/NACK bit clocked in from the slave.

A typical TWI master write sequence is illustrated in TWI master writing data to a slave on page 348. Occurrence 2 in the figure illustrates clock stretching performed by the TWI master following a SUSPEND task.

A SUSPENDED event indicates that the SUSPEND task has taken effect; this event can be used to synchronize the software.

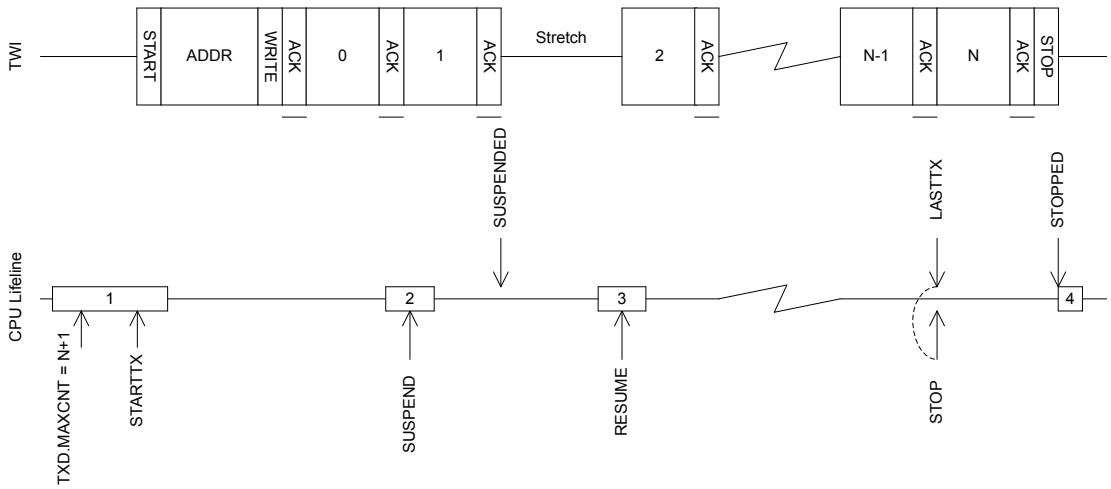



*Figure 131: TWI master writing data to a slave*

The TWI master will generate a LASTTX event when it starts to transmit the last byte, this is illustrated in TWI master writing data to a slave on page 348

The TWI master is stopped by triggering the STOP task, this task should be triggered during the transmission of the last byte to secure that the TWI will stop as fast as possible after sending the last byte. It is safe to use the shortcut between LASTTX and STOP to accomplish this.

Note that the TWI master does not stop by itself when the whole RAM buffer has been sent, or when an error occurs. The STOP task must be issued, through the use of a local or PPI shortcut, or in software as part of the error handler.

The TWI master cannot get stopped while it is suspended, so the STOP task has to be issued after the TWI master has been resumed.

### 6.25.3 Master read sequence

A TWI master read sequence is started by triggering the STARTRX task. After the STARTRX task has been triggered the TWI master will generate a start condition on the TWI bus, followed by clocking out the address and the READ/WRITE bit set to 1 (WRITE = 0, READ = 1). The address must match the address of the slave device that the master wants to read from. The READ/WRITE bit is followed by an ACK/NACK bit (ACK=0 or NACK = 1) generated by the slave.

After having sent the ACK bit the TWI slave will send data to the master using the clock generated by the master.

Data received will be stored in RAM at the address specified in the RXD.PTR register. The TWI master will generate an ACK after all but the last byte received from the slave. The TWI master will generate a NACK after the last byte received to indicate that the read sequence shall stop.

A typical TWI master read sequence is illustrated in The TWI master reading data from a slave on page 350. Occurrence 2 in the figure illustrates clock stretching performed by the TWI master following a SUSPEND task.

A SUSPENDED event indicates that the SUSPEND task has taken effect; this event can be used to synchronize the software.

The TWI master will generate a LASTRX event when it is ready to receive the last byte, this is illustrated in The TWI master reading data from a slave on page 350. If RXD.MAXCNT > 1 the LASTRX event is generated after sending the ACK of the previously received byte. If RXD.MAXCNT = 1 the LASTRX event is generated after receiving the ACK following the address and READ bit.

The TWI master is stopped by triggering the STOP task, this task must be triggered before the NACK bit is supposed to be transmitted. The STOP task can be triggered at any time during the reception of the last byte. It is safe to use the shortcut between LASTRX and STOP to accomplish this.

Note that the TWI master does not stop by itself when the RAM buffer is full, or when an error occurs. The STOP task must be issued, through the use of a local or PPI shortcut, or in software as part of the error handler.

The TWI master cannot get stopped while it is suspended, so the STOP task has to be issued after the TWI master has been resumed.

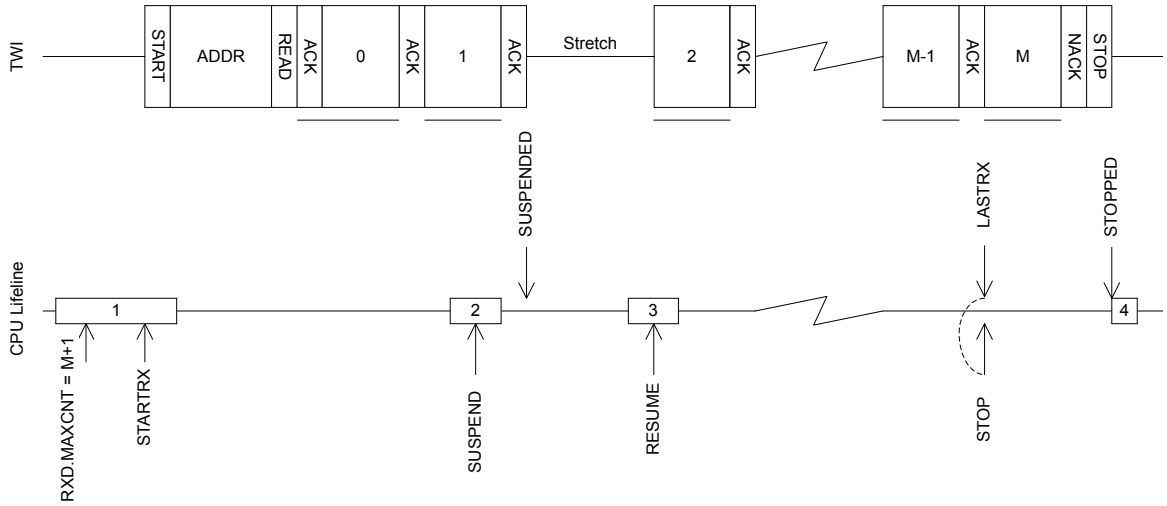



*Figure 132: The TWI master reading data from a slave*

## 6.25.4 Master repeated start sequence

A typical repeated start sequence is one in which the TWI master writes two bytes to the slave followed by reading four bytes from the slave. This example uses shortcuts to perform the simplest type of repeated start sequence, i.e. one write followed by one read. The same approach can be used to perform a repeated start sequence where the sequence is read followed by write.

The figure A repeated start sequence, where the TWI master writes two bytes followed by reading 4 bytes from the slave on page 350 illustrates this:

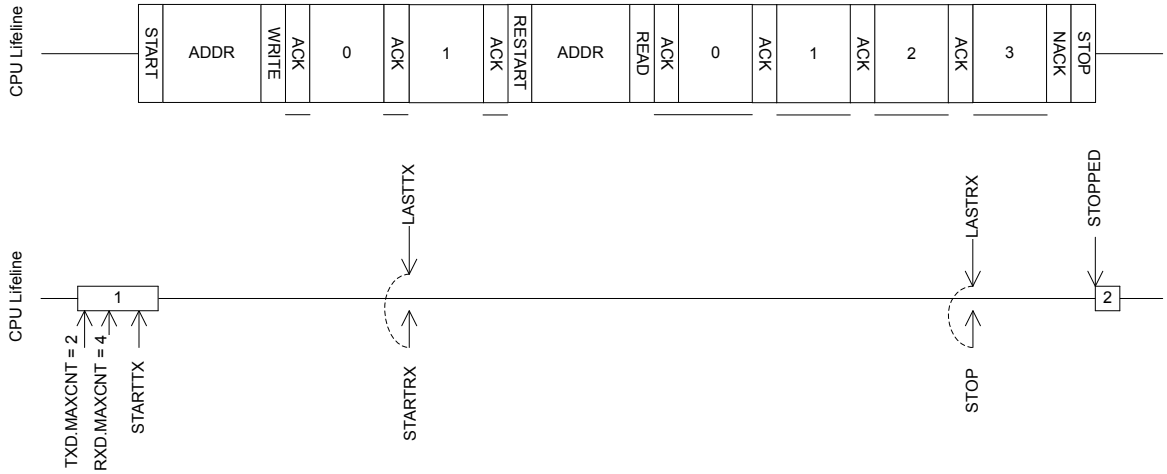



*Figure 133: A repeated start sequence, where the TWI master writes two bytes followed by reading 4 bytes from the slave*

If a more complex repeated start sequence is needed and the TWI firmware drive is serviced in a low priority interrupt it may be necessary to use the SUSPEND task and SUSPENDED event to guarantee that the correct tasks are generated at the correct time. This is illustrated in A double repeated start sequence using the SUSPEND task to secure safe operation in low priority interrupts on page 351.

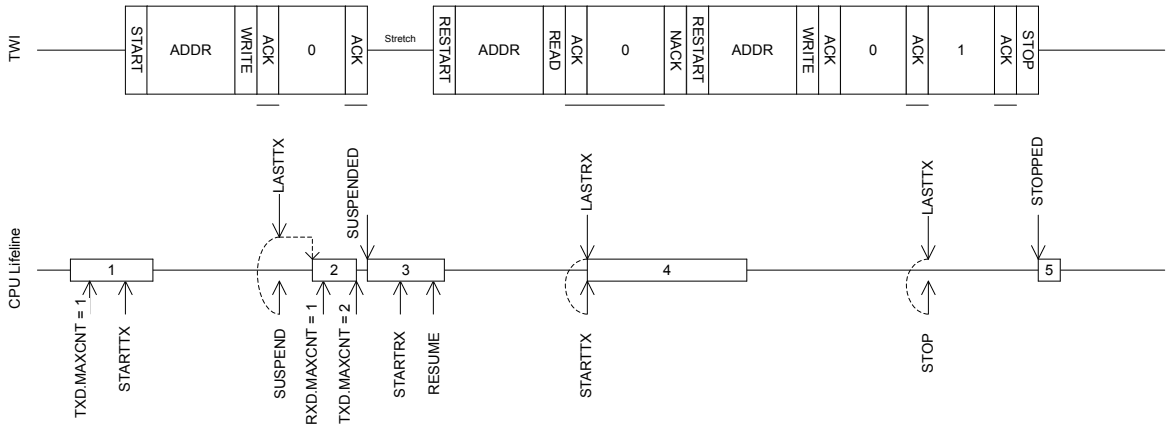



*Figure 134: A double repeated start sequence using the SUSPEND task to secure safe operation in low priority interrupts*

## 6.25.5 Low power

When putting the system in low power and the peripheral is not needed, lowest possible power consumption is achieved by stopping, and then disabling the peripheral.

The STOP task may not be always needed (the peripheral might already be stopped), but if it is sent, software shall wait until the STOPPED event was received as a response before disabling the peripheral through the ENABLE register.

## 6.25.6 Master mode pin configuration

The SCL and SDA signals associated with the TWI master are mapped to physical pins according to the configuration specified in the PSEL.SCL and PSEL.SDA registers respectively.

The PSEL.SCL and PSEL.SDA registers and their configurations are only used as long as the TWI master is enabled, and retained only as long as the device is in ON mode. When the peripheral is disabled, the pins will behave as regular GPIOs, and use the configuration in their respective OUT bit field and PIN_CNF[n] register. PSEL.SCL, PSEL.SDA must only be configured when the TWI master is disabled.

To secure correct signal levels on the pins used by the TWI master when the system is in OFF mode, and when the TWI master is disabled, these pins must be configured in the GPIO peripheral as described in GPIO configuration before enabling peripheral on page 351.

Only one peripheral can be assigned to drive a particular GPIO pin at a time. Failing to do so may result in unpredictable behavior.

| TWI master signal | TWI master pin | Direction | Output value | Drive strength |
|---|---|---|---|---|
| SCL | As specified in PSEL.SCL | Input | Not applicable | S0D1 |
| SDA | As specified in PSEL.SDA | Input | Not applicable | S0D1 |

*Table 104: GPIO configuration before enabling peripheral*

## 6.25.7 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40003000 | TWIM | TWIM0 | Two-wire interface master | |

*Table 105: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_STARTRX | 0x000 | Start TWI receive sequence |
| TASKS_STARTTX | 0x008 | Start TWI transmit sequence |
| TASKS_STOP | 0x014 | Stop TWI transaction. Must be issued while the TWI master is not suspended. |
| TASKS_SUSPEND | 0x01C | Suspend TWI transaction |
| TASKS_RESUME | 0x020 | Resume TWI transaction |
| EVENTS_STOPPED | 0x104 | TWI stopped |
| EVENTS_ERROR | 0x124 | TWI error |
| EVENTS_SUSPENDED | 0x148 | Last byte has been sent out after the SUSPEND task has been issued, TWI traffic is now suspended. |
| EVENTS_RXSTARTED | 0x14C | Receive sequence started |
| EVENTS_TXSTARTED | 0x150 | Transmit sequence started |
| EVENTS_LASTRX | 0x15C | Byte boundary, starting to receive the last byte |
| EVENTS_LASTTX | 0x160 | Byte boundary, starting to transmit the last byte |
| SHORTS | 0x200 | Shortcuts between local events and tasks |
| INTEN | 0x300 | Enable or disable interrupt |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| ERRORSRC | 0x4C4 | Error source |
| ENABLE | 0x500 | Enable TWIM |
| PSEL.SCL | 0x508 | Pin select for SCL signal |
| PSEL.SDA | 0x50C | Pin select for SDA signal |
| FREQUENCY | 0x524 | TWI frequency. Accuracy depends on the HFCLK source selected. |
| RXD.PTR | 0x534 | Data pointer |
| RXD.MAXCNT | 0x538 | Maximum number of bytes in receive buffer |
| RXD.AMOUNT | 0x53C | Number of bytes transferred in the last transaction |
| RXD.LIST | 0x540 | EasyDMA list type |
| TXD.PTR | 0x544 | Data pointer |
| TXD.MAXCNT | 0x548 | Maximum number of bytes in transmit buffer |
| TXD.AMOUNT | 0x54C | Number of bytes transferred in the last transaction |
| TXD.LIST | 0x550 | EasyDMA list type |
| ADDRESS | 0x588 | Address used in the TWI transfer |

*Table 106: Register overview*

## 6.25.7.1 TASKS_STARTRX

Address offset: 0x000

Start TWI receive sequence

| Bit number | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | A |
| **Reset 0x00000000** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_STARTRX | | | Start TWI receive sequence |
| | | | Trigger | 1 | Trigger task |

## 6.25.7.2 TASKS_STARTTX

Address offset: 0x008

Start TWI transmit sequence

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STARTTX | | | Start TWI transmit sequence |
| | | | Trigger | 1 | Trigger task |

### 6.25.7.3 TASKS_STOP

Address offset: 0x014

Stop TWI transaction. Must be issued while the TWI master is not suspended.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STOP | | | Stop TWI transaction. Must be issued while the TWI master is not suspended. |
| | | | Trigger | 1 | Trigger task |

### 6.25.7.4 TASKS_SUSPEND

Address offset: 0x01C

Suspend TWI transaction

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_SUSPEND | | | Suspend TWI transaction |
| | | | Trigger | 1 | Trigger task |

### 6.25.7.5 TASKS_RESUME

Address offset: 0x020

Resume TWI transaction

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_RESUME | | | Resume TWI transaction |
| | | | Trigger | 1 | Trigger task |

### 6.25.7.6 EVENTS_STOPPED

Address offset: 0x104

TWI stopped

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_STOPPED | | | TWI stopped |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.25.7.7 EVENTS_ERROR

Address offset: 0x124

TWI error

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_ERROR | | | TWI error |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.25.7.8 EVENTS_SUSPENDED

Address offset: 0x148

Last byte has been sent out after the SUSPEND task has been issued, TWI traffic is now suspended.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_SUSPENDED | | | Last byte has been sent out after the SUSPEND task has been issued, TWI traffic is now suspended. |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.25.7.9 EVENTS_RXSTARTED

Address offset: 0x14C

Receive sequence started

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_RXSTARTED | | | Receive sequence started |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.25.7.10 EVENTS_TXSTARTED

Address offset: 0x150

Transmit sequence started

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_TXSTARTED | | | Transmit sequence started |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.25.7.11 EVENTS_LASTRX

Address offset: 0x15C

Byte boundary, starting to receive the last byte

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_LASTRX | | | Byte boundary, starting to receive the last byte |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.25.7.12 EVENTS_LASTTX

Address offset: 0x160

Byte boundary, starting to transmit the last byte

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_LASTTX | | | Byte boundary, starting to transmit the last byte |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.25.7.13 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | LASTTX_STARTRX | | | Shortcut between event LASTTX and task STARTRX |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| B | RW | LASTTX_SUSPEND | | | Shortcut between event LASTTX and task SUSPEND |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| C | RW | LASTTX_STOP | | | Shortcut between event LASTTX and task STOP |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| D | RW | LASTRX_STARTTX | | | Shortcut between event LASTRX and task STARTTX |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| E | RW | LASTRX_SUSPEND | | | Shortcut between event LASTRX and task SUSPEND |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| F | RW | LASTRX_STOP | | | Shortcut between event LASTRX and task STOP |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |

## 6.25.7.14 INTEN

Address offset: 0x300

Enable or disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | J I H G F D A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | STOPPED | | | Enable or disable interrupt for event STOPPED |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| D | RW | ERROR | | | Enable or disable interrupt for event ERROR |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| F | RW | SUSPENDED | | | Enable or disable interrupt for event SUSPENDED |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| G | RW | RXSTARTED | | | Enable or disable interrupt for event RXSTARTED |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| H | RW | TXSTARTED | | | Enable or disable interrupt for event TXSTARTED |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| I | RW | LASTRX | | | Enable or disable interrupt for event LASTRX |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| J | RW | LASTTX | | | Enable or disable interrupt for event LASTTX |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |

## 6.25.7.15 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | J I H G F D A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | STOPPED | | | Write '1' to enable interrupt for event STOPPED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | ERROR | | | Write '1' to enable interrupt for event ERROR |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| F | RW | SUSPENDED | | | Write '1' to enable interrupt for event SUSPENDED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| G | RW | RXSTARTED | | | Write '1' to enable interrupt for event RXSTARTED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| H | RW | TXSTARTED | | | Write '1' to enable interrupt for event TXSTARTED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| I | RW | LASTRX | | | Write '1' to enable interrupt for event LASTRX |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| J | RW | LASTTX | | | Write '1' to enable interrupt for event LASTTX |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

### 6.25.7.16 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | J I H G F D A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | STOPPED | | | Write '1' to disable interrupt for event STOPPED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | ERROR | | | Write '1' to disable interrupt for event ERROR |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| F | RW | SUSPENDED | | | Write '1' to disable interrupt for event SUSPENDED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | J I H G F D A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| | | | Enabled | 1 | Read: Enabled |
| G | RW | RXSTARTED | | | Write '1' to disable interrupt for event RXSTARTED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| H | RW | TXSTARTED | | | Write '1' to disable interrupt for event TXSTARTED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| I | RW | LASTRX | | | Write '1' to disable interrupt for event LASTRX |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| J | RW | LASTTX | | | Write '1' to disable interrupt for event LASTTX |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.25.7.17 ERRORSRC

Address offset: 0x4C4

Error source

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | OVERRUN | | | Overrun error<br>A new byte was received before previous byte got transferred into RXD buffer. (Previous data is lost) |
| | | | NotReceived | 0 | Error did not occur |
| | | | Received | 1 | Error occurred |
| B | RW | ANACK | | | NACK received after sending the address (write '1' to clear) |
| | | | NotReceived | 0 | Error did not occur |
| | | | Received | 1 | Error occurred |
| C | RW | DNACK | | | NACK received after sending a data byte (write '1' to clear) |
| | | | NotReceived | 0 | Error did not occur |
| | | | Received | 1 | Error occurred |

## 6.25.7.18 ENABLE

Address offset: 0x500

Enable TWIM

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ENABLE | | | Enable or disable TWIM |
| | | | Disabled | 0 | Disable TWIM |
| | | | Enabled | 6 | Enable TWIM |

## 6.25.7.19 PSEL.SCL

Address offset: 0x508

Pin select for SCL signal

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.25.7.20 PSEL.SDA

Address offset: 0x50C

Pin select for SDA signal

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.25.7.21 FREQUENCY

Address offset: 0x524

TWI frequency. Accuracy depends on the HFCLK source selected.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x04000000** | | | | **0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | FREQUENCY | | | TWI master clock frequency |
| | | | K100 | 0x01980000 | 100 kbps |
| | | | K250 | 0x04000000 | 250 kbps |
| | | | K400 | 0x06400000 | 400 kbps |

## 6.25.7.22 RXD.PTR

Address offset: 0x534

Data pointer

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | PTR | | | Data pointer |

**Note:** See the memory chapter for details about which memories are available for EasyDMA.

## 6.25.7.23 RXD.MAXCNT

Address offset: 0x538

Maximum number of bytes in receive buffer

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | MAXCNT | | [1..0x3FFF] | Maximum number of bytes in receive buffer |

## 6.25.7.24 RXD.AMOUNT

Address offset: 0x53C

Number of bytes transferred in the last transaction

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | AMOUNT | | [1..0x3FFF] | Number of bytes transferred in the last transaction. In case of NACK error, includes the NACK'ed byte. |

## 6.25.7.25 RXD.LIST

Address offset: 0x540

EasyDMA list type

| Bit number | 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 |
|---|---|
| ID | A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | LIST | | | List type |
| | | | Disabled | 0 | Disable EasyDMA list |
| | | | ArrayList | 1 | Use array list |

## 6.25.7.26 TXD.PTR

Address offset: 0x544

Data pointer

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PTR | | | Data pointer |

**Note:** See the memory chapter for details about which memories are available for EasyDMA.

## 6.25.7.27 TXD.MAXCNT

Address offset: 0x548

Maximum number of bytes in transmit buffer

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | MAXCNT | | [1..0x3FFF] | Maximum number of bytes in transmit buffer |

## 6.25.7.28 TXD.AMOUNT

Address offset: 0x54C

Number of bytes transferred in the last transaction

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | AMOUNT | | [1..0x3FFF] | Number of bytes transferred in the last transaction. In case of NACK error, includes the NACK'ed byte. |

## 6.25.7.29 TXD.LIST

Address offset: 0x550

EasyDMA list type

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | LIST | | | List type |
| | | | Disabled | 0 | Disable EasyDMA list |
| | | | ArrayList | 1 | Use array list |

### 6.25.7.30 ADDRESS

Address offset: 0x588

Address used in the TWI transfer

| Bit number | 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 |
|---|---|
| ID | A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | ADDRESS | | | Address used in the TWI transfer |

## 6.25.8 Electrical specification

### 6.25.8.1 TWIM interface electrical specifications

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $f_{TWIM,SCL}$ | Bit rates for TWIM[30] | 100 | | 400 | kbps |
| $t_{TWIM,START}$ | Time from STARTRX/STARTTX task to transmission started | .. | .. | .. | µs |

### 6.25.8.2 Two Wire Interface Master (TWIM) timing specifications

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{TWIM,SU\_DAT}$ | Data setup time before positive edge on SCL – all modes | 300 | | | ns |
| $t_{TWIM,HD\_DAT}$ | Data hold time after negative edge on SCL – all modes | 500 | | | ns |
| $t_{TWIM,HD\_STA,100kbps}$ | TWIM master hold time for START and repeated START condition, 100 kbps | 10000 | | | ns |
| $t_{TWIM,HD\_STA,250kbps}$ | TWIM master hold time for START and repeated START condition, 250kbps | 4000 | | | ns |
| $t_{TWIM,HD\_STA,400kbps}$ | TWIM master hold time for START and repeated START condition, 400 kbps | 2500 | | | ns |
| $t_{TWIM,SU\_STO,100kbps}$ | TWIM master setup time from SCL high to STOP condition, 100 kbps | 5000 | | | ns |
| $t_{TWIM,SU\_STO,250kbps}$ | TWIM master setup time from SCL high to STOP condition, 250 kbps | 2000 | | | ns |
| $t_{TWIM,SU\_STO,400kbps}$ | TWIM master setup time from SCL high to STOP condition, 400 kbps | 1250 | | | ns |
| $t_{TWIM,BUF,100kbps}$ | TWIM master bus free time between STOP and START conditions, 100 kbps | 5800 | | | ns |
| $t_{TWIM,BUF,250kbps}$ | TWIM master bus free time between STOP and START conditions, 250 kbps | 2700 | | | ns |
| $t_{TWIM,BUF,400kbps}$ | TWIM master bus free time between STOP and START conditions, 400 kbps | 2100 | | | ns |

[30] High bit rates or stronger pull-ups may require GPIOs to be set as High Drive, see GPIO chapter for more details.

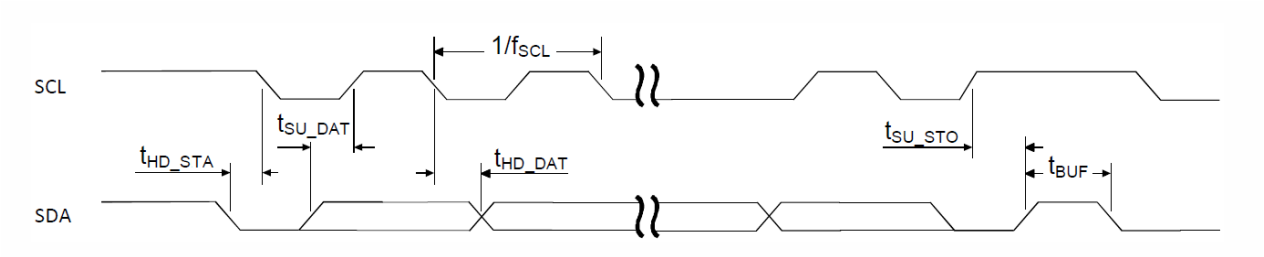



*Figure 135: TWIM timing diagram, 1 byte transaction*

### 6.25.9 Pullup resistor

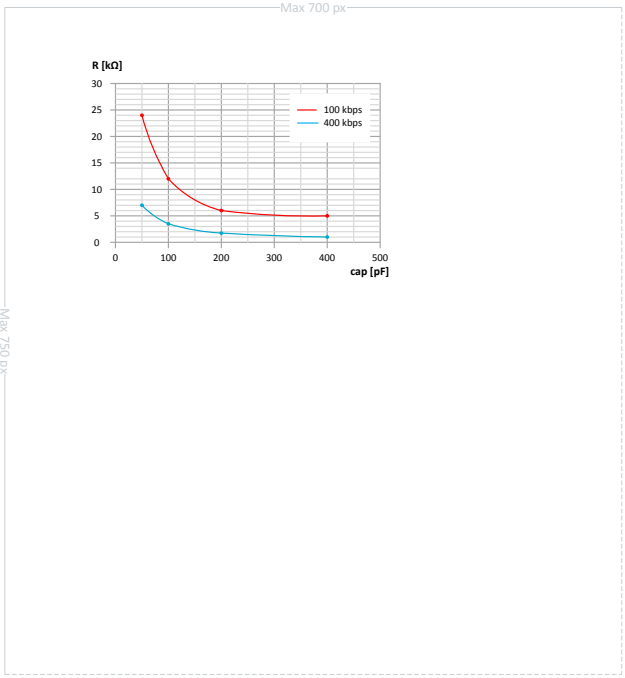



*Figure 136: Recommended TWIM pullup value vs. line capacitance*

- The I2C specification allows a line capacitance of 400 pF at most.
- The value of internal pullup resistor ($R_{PU}$) for nRF52811 can be found in GPIO — General purpose input/output on page 115.

## 6.26 TWIS — I²C compatible two-wire interface slave with EasyDMA

TWI slave with EasyDMA (TWIS) is compatible with I²C operating at 100 kHz and 400 kHz. The TWI transmitter and receiver implement EasyDMA.

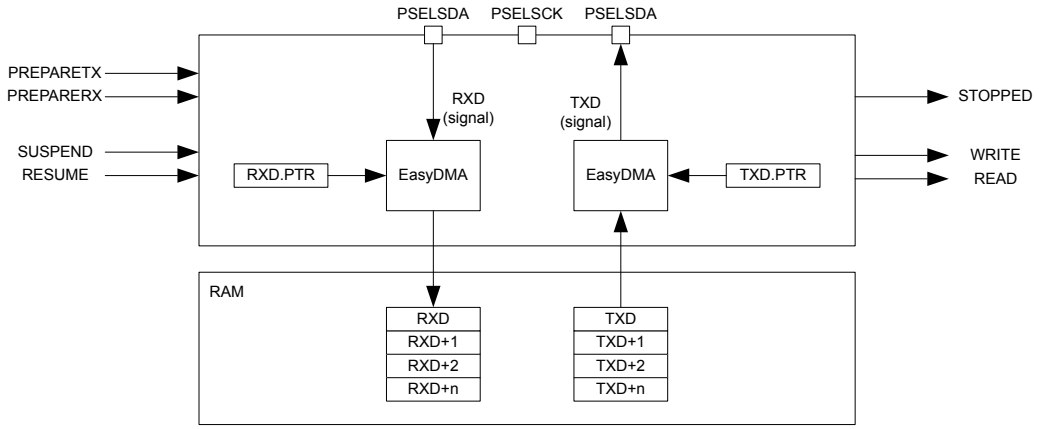



*Figure 137: TWI slave with EasyDMA*

A typical TWI setup consists of one master and one or more slaves. For an example, see A typical TWI setup comprising one master and three slaves on page 364. TWIS is only able to operate with a single master on the TWI bus.

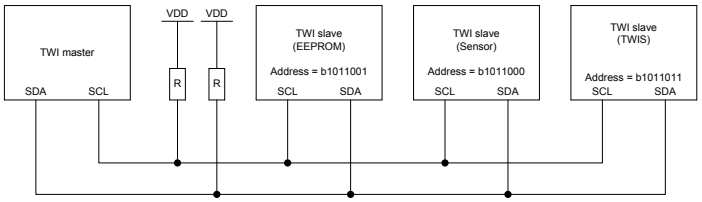



*Figure 138: A typical TWI setup comprising one master and three slaves*

The TWI slave state machine is illustrated in TWI slave state machine on page 365 and TWI slave state machine symbols on page 365 is explaining the different symbols used in the state machine.

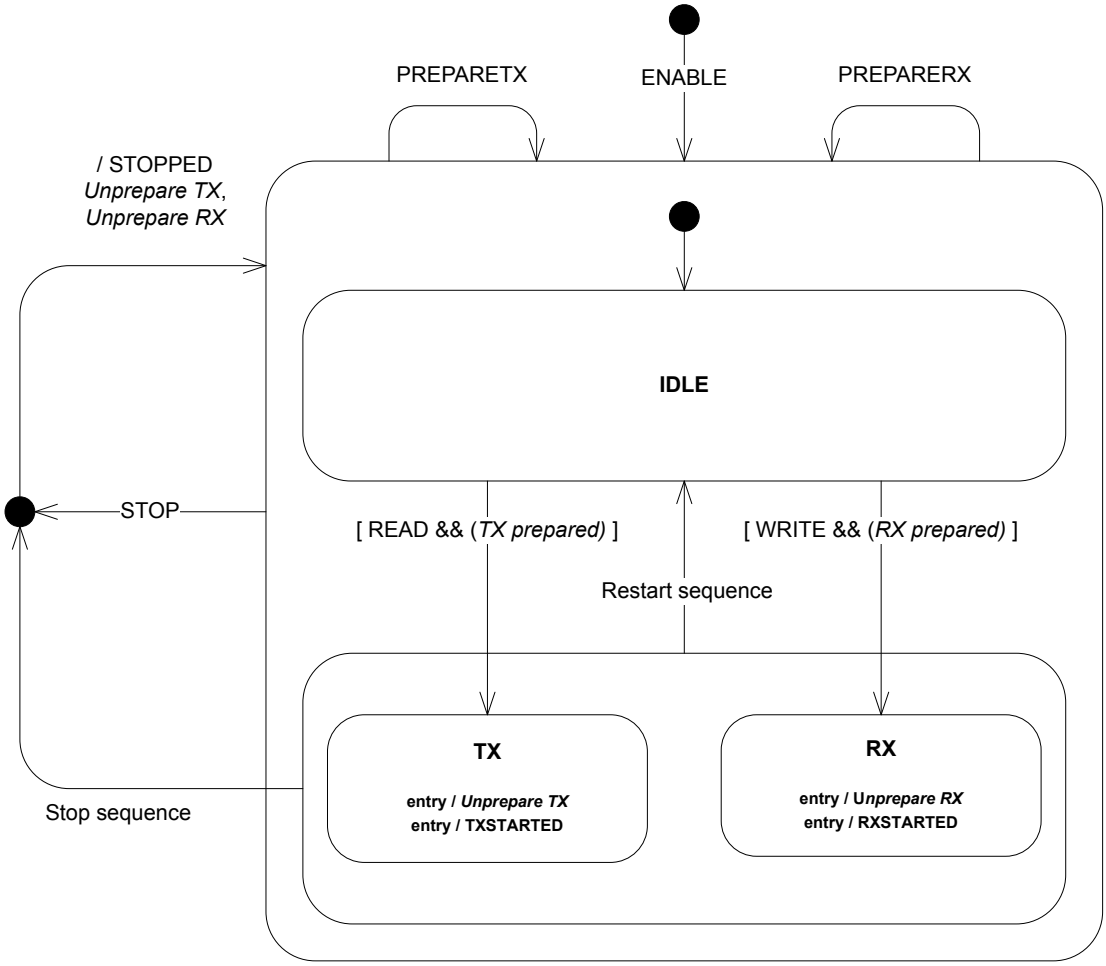



*Figure 139: TWI slave state machine*

| Symbol | Type | Description |
|---|---|---|
| ENABLE | Register | The TWI slave has been enabled via the ENABLE register |
| PREPARETX | Task | The TASKS_PREPARETX task has been triggered |
| STOP | Task | The TASKS_STOP task has been triggered |
| PREPARERX | Task | The TASKS_PREPARERX task has been triggered |
| STOPPED | Event | The EVENTS_STOPPED event was generated |
| RXSTARTED | Event | The EVENTS_RXSTARTED event was generated |
| TXSTARTED | Event | The EVENTS_TXSTARTED event was generated |
| TX prepared | Internal | Internal flag indicating that a TASKS_PREPARETX task has been triggered. This flag is not visible to the user. |
| RX prepared | Internal | Internal flag indicating that a TASKS_PREPARERX task has been triggered. This flag is not visible to the user. |
| Unprepare TX | Internal | Clears the internal 'TX prepared' flag until next TASKS_PREPARETX task. |
| Unprepare RX | Internal | Clears the internal 'RX prepared' flag until next TASKS_PREPARERX task. |
| Stop sequence | TWI protocol | A TWI stop sequence was detected |
| Restart sequence | TWI protocol | A TWI restart sequence was detected |

*Table 107: TWI slave state machine symbols*

The TWI slave supports clock stretching performed by the master.

The TWI slave operates in a low power mode while waiting for a TWI master to initiate a transfer. As long as the TWI slave is not addressed, it will remain in this low power mode.

To secure correct behaviour of the TWI slave, PSEL.SCL, PSEL.SDA, CONFIG and the ADDRESS[n] registers, must be configured prior to enabling the TWI slave through the ENABLE register. Similarly, changing these settings must be performed while the TWI slave is disabled. Failing to do so may result in unpredictable behaviour.

## 6.26.1 EasyDMA

The TWIS implements EasyDMA for accessing RAM without CPU involvement.

The TWIS peripheral implements the following EasyDMA channels:

| Channel | Type | Register Cluster |
|---|---|---|
| TXD | READER | TXD |
| RXD | WRITER | RXD |

*Table 108: TWIS EasyDMA Channels*

For detailed information regarding the use of EasyDMA, see EasyDMA on page 35.

The STOPPED event indicates that EasyDMA has finished accessing the buffer in RAM.

## 6.26.2 TWI slave responding to a read command

Before the TWI slave can respond to a read command the TWI slave must be configured correctly and enabled via the ENABLE register. When enabled the TWI slave will be in its IDLE state where it will consume $I_{IDLE}$ .

A read command is started when the TWI master generates a start condition on the TWI bus, followed by clocking out the address and the READ/WRITE bit set to 1 (WRITE=0, READ=1). The READ/WRITE bit is followed by an ACK/NACK bit (ACK=0 or NACK=1) response from the TWI slave.

The TWI slave is able to listen for up to two addresses at the same time. Which addresses to listen for is configured in the ADDRESS registers and the CONFIG register.

The TWI slave will only acknowledge (ACK) the read command if the address presented by the master matches one of the addresses the slave is configured to listen for. The TWI slave will generate a READ event when it acknowledges the read command.

The TWI slave is only able to detect a read command from the IDLE state.

The TWI slave will set an internal 'TX prepared' flag when the PREPARETX task is triggered.

When the read command is received the TWI slave will enter the TX state if the internal 'TX prepared' flag is set.

If the internal 'TX prepared' flag is not set when the read command is received, the TWI slave will stretch the master's clock until the PREPARETX task is triggered and the internal 'TX prepared' flag is set.

The TWI slave will generate the TXSTARTED event and clear the 'TX prepared' flag ('unprepare TX') when it enters the TX state. In this state the TWI slave will send the data bytes found in the transmit buffer to the master using the master's clock. The TWI slave will consume $I_{TX}$ in this mode.

The TWI slave will go back to the IDLE state if the TWI slave receives a restart command when it is in the TX state.

The TWI slave is stopped when it receives the stop condition from the TWI master. A STOPPED event will be generated when the transaction has stopped. The TWI slave will clear the 'TX prepared' flag ('unprepare TX') and go back to the IDLE state when it has stopped.

The transmit buffer is located in RAM at the address specified in the TXD.PTR register. The TWI slave will only be able to send TXD.MAXCNT bytes from the transmit buffer for each transaction. If the TWI master forces the slave to send more than TXD.MAXCNT bytes, the slave will send the byte specified in the ORC register to the master instead. If this happens, an ERROR event will be generated.

The EasyDMA configuration registers, see TXD.PTR etc., are latched when the TXSTARTED event is generated.

The TWI slave can be forced to stop by triggering the STOP task. A STOPPED event will be generated when the TWI slave has stopped. The TWI slave will clear the 'TX prepared' flag and go back to the IDLE state when it has stopped, see also Terminating an ongoing TWI transaction on page 369.

Each byte sent from the slave will be followed by an ACK/NACK bit sent from the master. The TWI master will generate a NACK following the last byte that it wants to receive to tell the slave to release the bus so that the TWI master can generate the stop condition. The TXD.AMOUNT register can be queried after a transaction to see how many bytes were sent.

A typical TWI slave read command response is illustrated in The TWI slave responding to a read command on page 367. Occurrence 2 in the figure illustrates clock stretching performed by the TWI slave following a SUSPEND task.

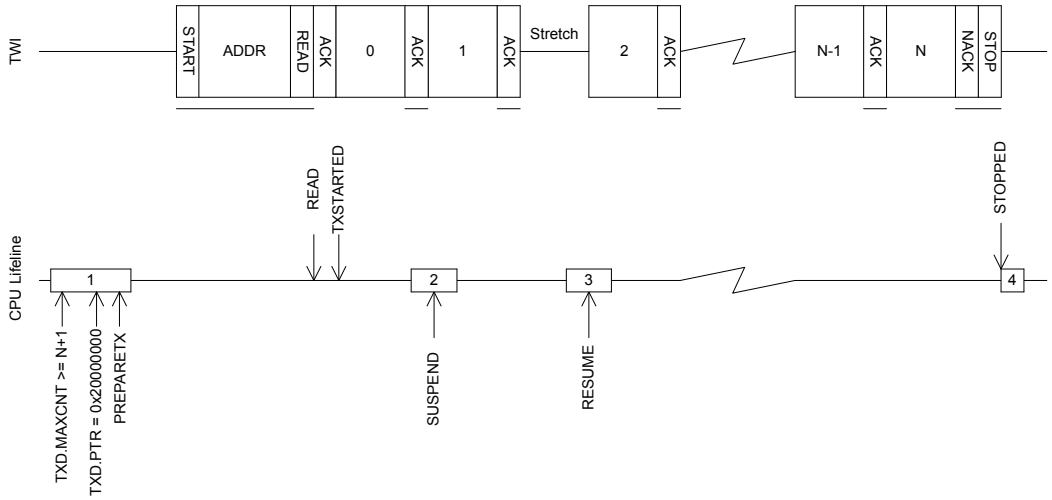



*Figure 140: The TWI slave responding to a read command*

## 6.26.3 TWI slave responding to a write command

Before the TWI slave can respond to a write command the TWI slave must be configured correctly and enabled via the ENABLE register. When enabled the TWI slave will be in its IDLE state where it will consume $I_{IDLE}$.

A write command is started when the TWI master generates a start condition on the TWI bus, followed by clocking out the address and the READ/WRITE bit set to 0 (WRITE=0, READ=1). The READ/WRITE bit is followed by an ACK/NACK bit (ACK=0 or NACK=1) response from the slave.

The TWI slave is able to listen for up to two addresses at the same time. Which addresses to listen for is configured in the ADDRESS registers and the CONFIG register.

The TWI slave will only acknowledge (ACK) the write command if the address presented by the master matches one of the addresses the slave is configured to listen for. The TWI slave will generate a WRITE event if it acknowledges the write command.

The TWI slave is only able to detect a write command from the IDLE state.

The TWI slave will set an internal 'RX prepared' flag when the PREPARERX task is triggered.

When the write command is received the TWI slave will enter the RX state if the internal 'RX prepared' flag is set.

If the internal 'RX prepared' flag is not set when the write command is received, the TWI slave will stretch the master's clock until the PREPARERX task is triggered and the internal 'RX prepared' flag is set.

The TWI slave will generate the RXSTARTED event and clear the internal 'RX prepared' flag ('unprepare RX') when it enters the RX state. In this state the TWI slave will be able to receive the bytes sent by the TWI master. The TWI slave will consume $I_{RX}$ in this mode.

The TWI slave will go back to the IDLE state if the TWI slave receives a restart command when it is in the RX state.

The TWI slave is stopped when it receives the stop condition from the TWI master. A STOPPED event will be generated when the transaction has stopped. The TWI slave will clear the internal 'RX prepared' flag ('unprepare RX') and go back to the IDLE state when it has stopped.

The receive buffer is located in RAM at the address specified in the TXD.PTR register. The TWI slave will only be able to receive as many bytes as specified in the RXD.MAXCNT register. If the TWI master tries to send more bytes to the slave than the slave is able to receive,these bytes will be discarded and the bytes will be NACKed by the slave. If this happens, an ERROR event will be generated.

The EasyDMA configuration registers, see RXD.PTR etc., are latched when the RXSTARTED event is generated.

The TWI slave can be forced to stop by triggering the STOP task. A STOPPED event will be generated when the TWI slave has stopped. The TWI slave will clear the internal 'RX prepared' flag and go back to the IDLE state when it has stopped, see also Terminating an ongoing TWI transaction on page 369.

The TWI slave will generate an ACK after every byte received from the master. The RXD.AMOUNT register can be queried after a transaction to see how many bytes were received.

A typical TWI slave write command response is illustrated in The TWI slave responding to a write command on page 368. Occurrence 2 in the figure illustrates clock stretching performed by the TWI slave following a SUSPEND task.

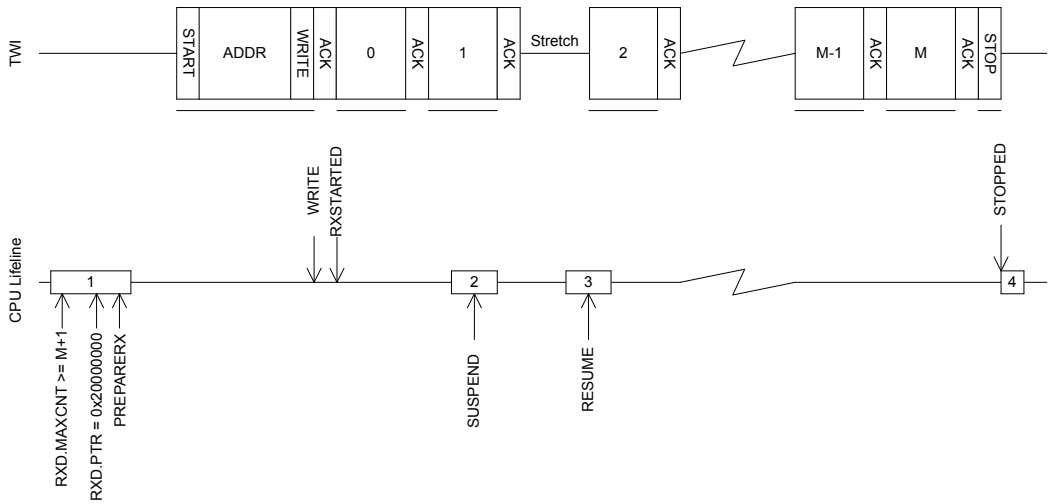



*Figure 141: The TWI slave responding to a write command*

## 6.26.4 Master repeated start sequence

An example of a repeated start sequence is one in which the TWI master writes two bytes to the slave followed by reading four bytes from the slave.

This is illustrated in A repeated start sequence, where the TWI master writes two bytes followed by reading four bytes from the slave on page 369.

It is here assumed that the receiver does not know in advance what the master wants to read, and that this information is provided in the first two bytes received in the write part of the repeated start sequence. To guarantee that the CPU is able to process the received data before the TWI slave starts to reply to the read command, the SUSPEND task is triggered via a shortcut from the READ event generated when the read command is received. When the CPU has processed the incoming data and prepared the correct data response, the CPU will resume the transaction by triggering the RESUME task.

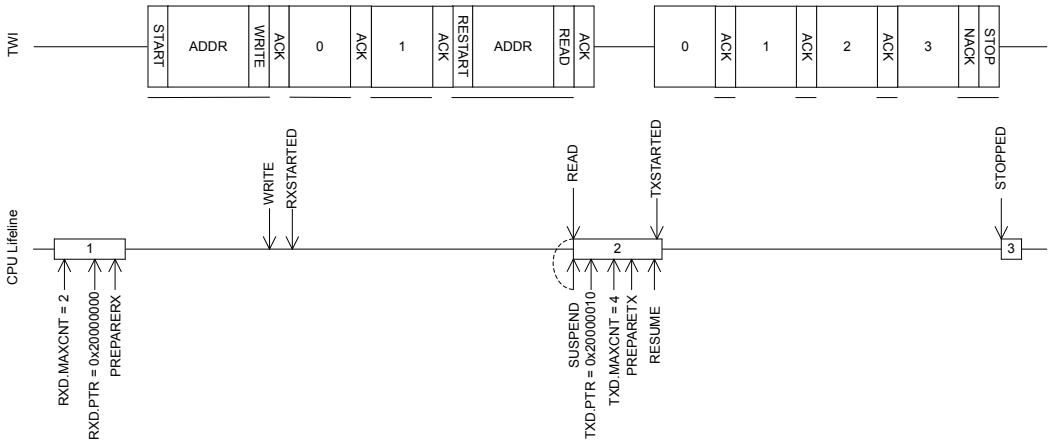



*Figure 142: A repeated start sequence, where the TWI master writes two bytes followed by reading four bytes from the slave*

## 6.26.5 Terminating an ongoing TWI transaction

In some situations, e.g. if the external TWI master is not responding correctly, it may be required to terminate an ongoing transaction.

This can be achieved by triggering the STOP task. In this situation a STOPPED event will be generated when the TWI has stopped independent of whether or not a STOP condition has been generated on the TWI bus. The TWI slave will release the bus when it has stopped and go back to its IDLE state.

## 6.26.6 Low power

When putting the system in low power and the peripheral is not needed, lowest possible power consumption is achieved by stopping, and then disabling the peripheral.

The STOP task may not be always needed (the peripheral might already be stopped), but if it is sent, software shall wait until the STOPPED event was received as a response before disabling the peripheral through the ENABLE register.

## 6.26.7 Slave mode pin configuration

The SCL and SDA signals associated with the TWI slave are mapped to physical pins according to the configuration specified in the PSEL.SCL and PSEL.SDA registers respectively.

The PSEL.SCL and PSEL.SDA registers and their configurations are only used as long as the TWI slave is enabled, and retained only as long as the device is in ON mode. When the peripheral is disabled, the pins will behave as regular GPIOs, and use the configuration in their respective OUT bit field and PIN_CNF[n] register. PSEL.SCL and PSEL.SDA must only be configured when the TWI slave is disabled.

To secure correct signal levels on the pins used by the TWI slave when the system is in OFF mode, and when the TWI slave is disabled, these pins must be configured in the GPIO peripheral as described in GPIO configuration before enabling peripheral on page 370.

Only one peripheral can be assigned to drive a particular GPIO pin at a time. Failing to do so may result in unpredictable behavior.

| TWI slave signal | TWI slave pin | Direction | Output value | Drive strength |
|---|---|---|---|---|
| SCL | As specified in PSEL.SCL | Input | Not applicable | S0D1 |
| SDA | As specified in PSEL.SDA | Input | Not applicable | S0D1 |

*Table 109: GPIO configuration before enabling peripheral*

## 6.26.8 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40003000 | TWIS | TWIS0 | Two-wire interface slave | |

*Table 110: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_STOP | 0x014 | Stop TWI transaction |
| TASKS_SUSPEND | 0x01C | Suspend TWI transaction |
| TASKS_RESUME | 0x020 | Resume TWI transaction |
| TASKS_PREPARERX | 0x030 | Prepare the TWI slave to respond to a write command |
| TASKS_PREPARETX | 0x034 | Prepare the TWI slave to respond to a read command |
| EVENTS_STOPPED | 0x104 | TWI stopped |
| EVENTS_ERROR | 0x124 | TWI error |
| EVENTS_RXSTARTED | 0x14C | Receive sequence started |
| EVENTS_TXSTARTED | 0x150 | Transmit sequence started |
| EVENTS_WRITE | 0x164 | Write command received |
| EVENTS_READ | 0x168 | Read command received |
| SHORTS | 0x200 | Shortcuts between local events and tasks |
| INTEN | 0x300 | Enable or disable interrupt |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| ERRORSRC | 0x4D0 | Error source |
| MATCH | 0x4D4 | Status register indicating which address had a match |
| ENABLE | 0x500 | Enable TWIS |
| PSEL.SCL | 0x508 | Pin select for SCL signal |
| PSEL.SDA | 0x50C | Pin select for SDA signal |
| RXD.PTR | 0x534 | RXD Data pointer |
| RXD.MAXCNT | 0x538 | Maximum number of bytes in RXD buffer |
| RXD.AMOUNT | 0x53C | Number of bytes transferred in the last RXD transaction |
| RXD.LIST | 0x540 | EasyDMA list type |
| TXD.PTR | 0x544 | TXD Data pointer |
| TXD.MAXCNT | 0x548 | Maximum number of bytes in TXD buffer |
| TXD.AMOUNT | 0x54C | Number of bytes transferred in the last TXD transaction |
| TXD.LIST | 0x550 | EasyDMA list type |
| ADDRESS[0] | 0x588 | TWI slave address 0 |
| ADDRESS[1] | 0x58C | TWI slave address 1 |
| CONFIG | 0x594 | Configuration register for the address match mechanism |
| ORC | 0x5C0 | Over-read character. Character sent out in case of an over-read of the transmit buffer. |

*Table 111: Register overview*

### 6.26.8.1 TASKS_STOP

Address offset: 0x014

Stop TWI transaction

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STOP | | | Stop TWI transaction |
| | | | Trigger | 1 | Trigger task |

### 6.26.8.2 TASKS_SUSPEND

Address offset: 0x01C

Suspend TWI transaction

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_SUSPEND | | | Suspend TWI transaction |
| | | | Trigger | 1 | Trigger task |

### 6.26.8.3 TASKS_RESUME

Address offset: 0x020

Resume TWI transaction

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_RESUME | | | Resume TWI transaction |
| | | | Trigger | 1 | Trigger task |

### 6.26.8.4 TASKS_PREPARERX

Address offset: 0x030

Prepare the TWI slave to respond to a write command

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_PREPARERX | | | Prepare the TWI slave to respond to a write command |
| | | | Trigger | 1 | Trigger task |

### 6.26.8.5 TASKS_PREPARETX

Address offset: 0x034

Prepare the TWI slave to respond to a read command

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_PREPARETX | | | Prepare the TWI slave to respond to a read command |
| | | | Trigger | 1 | Trigger task |

## 6.26.8.6 EVENTS_STOPPED

Address offset: 0x104

TWI stopped

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_STOPPED | | | TWI stopped |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.26.8.7 EVENTS_ERROR

Address offset: 0x124

TWI error

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_ERROR | | | TWI error |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.26.8.8 EVENTS_RXSTARTED

Address offset: 0x14C

Receive sequence started

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_RXSTARTED | | | Receive sequence started |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.26.8.9 EVENTS_TXSTARTED

Address offset: 0x150

Transmit sequence started

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_TXSTARTED | | | Transmit sequence started |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.26.8.10 EVENTS_WRITE

Address offset: 0x164

Write command received

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_WRITE | | | Write command received |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.26.8.11 EVENTS_READ

Address offset: 0x168

Read command received

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_READ | | | Read command received |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.26.8.12 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | WRITE_SUSPEND | | | Shortcut between event WRITE and task SUSPEND |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| B | RW | READ_SUSPEND | | | Shortcut between event READ and task SUSPEND |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |

## 6.26.8.13 INTEN

Address offset: 0x300

Enable or disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | H G F E B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | STOPPED | | | Enable or disable interrupt for event STOPPED |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| B | RW | ERROR | | | Enable or disable interrupt for event ERROR |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| E | RW | RXSTARTED | | | Enable or disable interrupt for event RXSTARTED |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| F | RW | TXSTARTED | | | Enable or disable interrupt for event TXSTARTED |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| G | RW | WRITE | | | Enable or disable interrupt for event WRITE |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| H | RW | READ | | | Enable or disable interrupt for event READ |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |

## 6.26.8.14 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | H G F E B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | STOPPED | | | Write '1' to enable interrupt for event STOPPED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | ERROR | | | Write '1' to enable interrupt for event ERROR |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E | RW | RXSTARTED | | | Write '1' to enable interrupt for event RXSTARTED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| F | RW | TXSTARTED | | | Write '1' to enable interrupt for event TXSTARTED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | H G F E B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| G | RW | WRITE | | | Write '1' to enable interrupt for event WRITE |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| H | RW | READ | | | Write '1' to enable interrupt for event READ |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.26.8.15 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | H G F E B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | STOPPED | | | Write '1' to disable interrupt for event STOPPED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | ERROR | | | Write '1' to disable interrupt for event ERROR |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E | RW | RXSTARTED | | | Write '1' to disable interrupt for event RXSTARTED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| F | RW | TXSTARTED | | | Write '1' to disable interrupt for event TXSTARTED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| G | RW | WRITE | | | Write '1' to disable interrupt for event WRITE |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| H | RW | READ | | | Write '1' to disable interrupt for event READ |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.26.8.16 ERRORSRC

Address offset: 0x4D0

Error source

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | OVERFLOW | | | RX buffer overflow detected, and prevented |
| | | | NotDetected | 0 | Error did not occur |
| | | | Detected | 1 | Error occurred |
| B | RW | DNACK | | | NACK sent after receiving a data byte |
| | | | NotReceived | 0 | Error did not occur |
| | | | Received | 1 | Error occurred |
| C | RW | OVERREAD | | | TX buffer over-read detected, and prevented |
| | | | NotDetected | 0 | Error did not occur |
| | | | Detected | 1 | Error occurred |

## 6.26.8.17 MATCH

Address offset: 0x4D4

Status register indicating which address had a match

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | MATCH | | [0..1] | Which of the addresses in {ADDRESS} matched the incoming address |

## 6.26.8.18 ENABLE

Address offset: 0x500

Enable TWIS

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ENABLE | | | Enable or disable TWIS |
| | | | Disabled | 0 | Disable TWIS |
| | | | Enabled | 9 | Enable TWIS |

## 6.26.8.19 PSEL.SCL

Address offset: 0x508

Pin select for SCL signal

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.26.8.20 PSEL.SDA

Address offset: 0x50C

Pin select for SDA signal

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.26.8.21 RXD.PTR

Address offset: 0x534

RXD Data pointer

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PTR | | | RXD Data pointer |

**Note:** See the memory chapter for details about which memories are available for EasyDMA.

## 6.26.8.22 RXD.MAXCNT

Address offset: 0x538

Maximum number of bytes in RXD buffer

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | MAXCNT | | [1..0x3FFF] | Maximum number of bytes in RXD buffer |

## 6.26.8.23 RXD.AMOUNT

Address offset: 0x53C

Number of bytes transferred in the last RXD transaction

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | AMOUNT | | [1..0x3FFF] | Number of bytes transferred in the last RXD transaction |

## 6.26.8.24 RXD.LIST

Address offset: 0x540

EasyDMA list type

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | LIST | | | List type |
| | | | Disabled | 0 | Disable EasyDMA list |
| | | | ArrayList | 1 | Use array list |

## 6.26.8.25 TXD.PTR

Address offset: 0x544

TXD Data pointer

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PTR | | | TXD Data pointer |

**Note:** See the memory chapter for details about which memories are available for EasyDMA.

## 6.26.8.26 TXD.MAXCNT

Address offset: 0x548

Maximum number of bytes in TXD buffer

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | MAXCNT | | [1..0x3FFF] | Maximum number of bytes in TXD buffer |

## 6.26.8.27 TXD.AMOUNT

Address offset: 0x54C

Number of bytes transferred in the last TXD transaction

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | AMOUNT | | [1..0x3FFF] | Number of bytes transferred in the last TXD transaction |

## 6.26.8.28 TXD.LIST

Address offset: 0x550

EasyDMA list type

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | LIST | | | List type |
| | | | Disabled | 0 | Disable EasyDMA list |
| | | | ArrayList | 1 | Use array list |

## 6.26.8.29 ADDRESS[n] (n=0..1)

Address offset: 0x588 + (n × 0x4)

TWI slave address n

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ADDRESS | | | TWI slave address |

## 6.26.8.30 CONFIG

Address offset: 0x594

Configuration register for the address match mechanism

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | B A |
| **Reset 0x00000001** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A-B | RW | ADDRESS[i] (i=0..1) | | | Enable or disable address matching on ADDRESS[i] |
| | | | Disabled | 0 | Disabled |
| | | | Enabled | 1 | Enabled |

## 6.26.8.31 ORC

Address offset: 0x5C0

Over-read character. Character sent out in case of an over-read of the transmit buffer.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ORC | | | Over-read character. Character sent out in case of an over-read of the transmit buffer. |

## 6.26.9 Electrical specification

### 6.26.9.1 TWIS slave timing specifications

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $f_{TWIS,SCL}$ | Bit rates for TWIS[31] | 100 | | 400 | kbps |
| $t_{TWIS,START}$ | Time from PREPARERX/PREPARETX task to ready to receive/ transmit | | 1.5 | | µs |
| $t_{TWIS,SU\_DAT}$ | Data setup time before positive edge on SCL – all modes | 300 | | | ns |
| $t_{TWIS,HD\_DAT}$ | Data hold time after negative edge on SCL – all modes | 500 | | | ns |
| $t_{TWIS,HD\_STA,100kbps}$ | TWI slave hold time from for START condition (SDA low to SCL low), 100 kbps | 5200 | | | ns |
| $t_{TWIS,HD\_STA,400kbps}$ | TWI slave hold time from for START condition (SDA low to SCL low), 400 kbps | 1300 | | | ns |
| $t_{TWIS,SU\_STO,100kbps}$ | TWI slave setup time from SCL high to STOP condition, 100 kbps | 5200 | | | ns |
| $t_{TWIS,SU\_STO,400kbps}$ | TWI slave setup time from SCL high to STOP condition, 400 kbps | 1300 | | | ns |
| $t_{TWIS,BUF,100kbps}$ | TWI slave bus free time between STOP and START conditions, 100 kbps | | 4700 | | ns |
| $t_{TWIS,BUF,400kbps}$ | TWI slave bus free time between STOP and START conditions, 400 kbps | | 1300 | | ns |

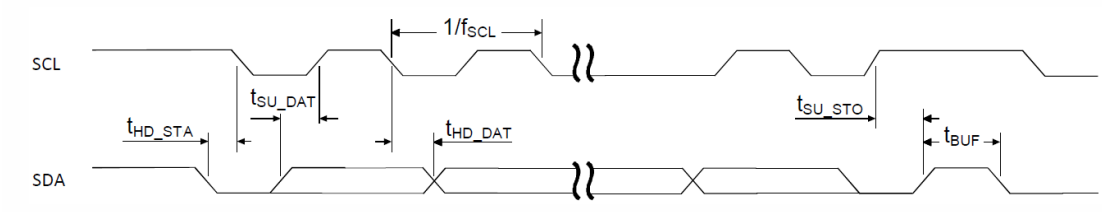



*Figure 143: TWIS timing diagram, 1 byte transaction*

# 6.27 UART — Universal asynchronous receiver/ transmitter

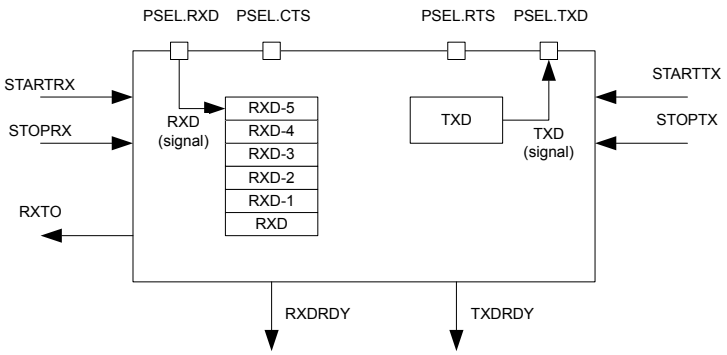



*Figure 144: UART configuration*

[31] High bit rates or stronger pull-ups may require GPIOs to be set as High Drive, see GPIO chapter for more details.

## 6.27.1 Functional description

Listed here are the main features of UART.

The UART implements support for the following features:

- Full-duplex operation
- Automatic flow control
- Parity checking and generation for the 9$^{th}$ data bit

As illustrated in UART configuration on page 380, the UART uses the TXD and RXD registers directly to transmit and receive data. The UART uses one stop bit.

**Note:** External crystal oscillator must be enabled to obtain sufficient clock accuracy for stable communication. See CLOCK — Clock control on page 61 for more information.

## 6.27.2 Pin configuration

The different signals RXD, CTS (Clear To Send, active low), RTS (Request To Send, active low), and TXD associated with the UART are mapped to physical pins according to the configuration specified in the PSEL.RXD, PSEL.CTS, PSEL.RTS, and PSEL.TXD registers respectively.

If the CONNECT field of a PSEL.xxx register is set to Disconnected, the associated UART signal will not be connected to any physical pin. The PSEL.RXD, PSEL.CTS, PSEL.RTS, and PSEL.TXD registers and their configurations are only used as long as the UART is enabled, and retained only for the duration the device is in ON mode. PSEL.RXD, PSEL.CTS, PSEL.RTS and PSEL.TXD must only be configured when the UART is disabled.

To secure correct signal levels on the pins by the UART when the system is in OFF mode, the pins must be configured in the GPIO peripheral as described in Pin configuration on page 381.

Only one peripheral can be assigned to drive a particular GPIO pin at a time. Failing to do so may result in unpredictable behavior.

| UART pin | Direction | Output value |
|---|---|---|
| RXD | Input | Not applicable |
| CTS | Input | Not applicable |
| RTS | Output | 1 |
| TXD | Output | 1 |

*Table 112: GPIO configuration*

## 6.27.3 Shared resources

The UART shares registers and other resources with other peripherals that have the same ID as the UART.

Therefore, you must disable all peripherals that have the same ID as the UART before the UART can be configured and used. Disabling a peripheral that has the same ID as the UART will not reset any of the registers that are shared with the UART. It is therefore important to configure all relevant UART registers explicitly to ensure that it operates correctly.

See the Instantiation table in Instantiation on page 18 for details on peripherals and their IDs.

## 6.27.4 Transmission

A UART transmission sequence is started by triggering the STARTTX task.

Bytes are transmitted by writing to the TXD register. When a byte has been successfully transmitted the UART will generate a TXDRDY event after which a new byte can be written to the TXD register. A UART transmission sequence is stopped immediately by triggering the STOPTX task.

If flow control is enabled a transmission will be automatically suspended when CTS is deactivated and resumed when CTS is activated again, as illustrated in UART transmission on page 382. A byte that is in transmission when CTS is deactivated will be fully transmitted before the transmission is suspended. For more information, see Suspending the UART on page 383.

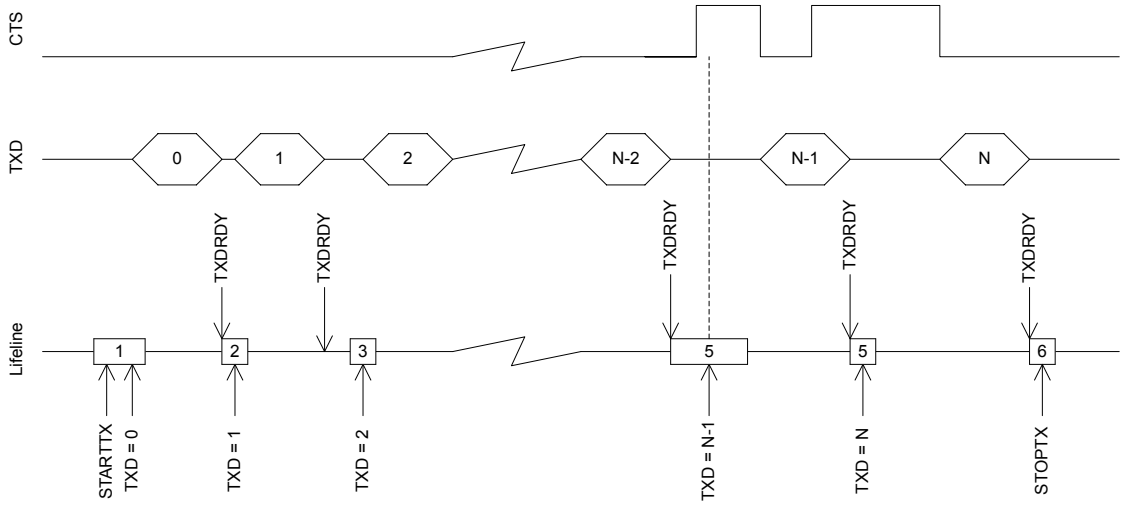



*Figure 145: UART transmission*

## 6.27.5 Reception

A UART reception sequence is started by triggering the STARTRX task.

The UART receiver chain implements a FIFO capable of storing six incoming RXD bytes before data is overwritten. Bytes are extracted from this FIFO by reading the RXD register. When a byte is extracted from the FIFO a new byte pending in the FIFO will be moved to the RXD register. The UART will generate an RXDRDY event every time a new byte is moved to the RXD register.

When flow control is enabled, the UART will deactivate the RTS signal when there is only space for four more bytes in the receiver FIFO. The counterpart transmitter is therefore able to send up to four bytes after the RTS signal is deactivated before data is being overwritten. To prevent overwriting data in the FIFO, the counterpart UART transmitter must therefore make sure to stop transmitting data within four bytes after the RTS line is deactivated.

The RTS signal will first be activated again when the FIFO has been emptied, that is, when all bytes in the FIFO have been read by the CPU, see UART reception on page 383.

The RTS signal will also be deactivated when the receiver is stopped through the STOPRX task as illustrated in UART reception on page 383. The UART is able to receive four to five additional bytes if they are sent in succession immediately after the RTS signal has been deactivated. This is possible because the UART is, even after the STOPRX task is triggered, able to receive bytes for an extended period of time dependent on the configured baud rate. The UART will generate a receiver timeout event (RXTO) when this period has elapsed.

To prevent loss of incoming data the RXD register must only be read one time following every RXDRDY event.

To secure that the CPU can detect all incoming RXDRDY events through the RXDRDY event register, the RXDRDY event register must be cleared before the RXD register is read. The reason for this is that the UART is allowed to write a new byte to the RXD register, and therefore can also generate a new event, immediately after the RXD register is read (emptied) by the CPU.

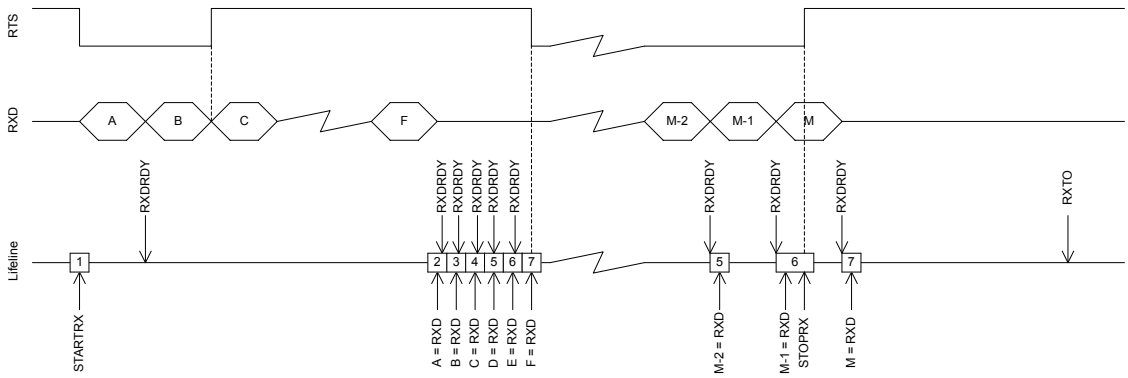



*Figure 146: UART reception*

As indicated in occurrence 2 in the figure, the RXDRDY event associated with byte B is generated first after byte A has been extracted from RXD.

## 6.27.6 Suspending the UART

The UART can be suspended by triggering the SUSPEND task.

SUSPEND will affect both the UART receiver and the UART transmitter, i.e. the transmitter will stop transmitting and the receiver will stop receiving. UART transmission and reception can be resumed, after being suspended, by triggering STARTTX and STARTRX respectively.

Following a SUSPEND task, an ongoing TXD byte transmission will be completed before the UART is suspended.

When the SUSPEND task is triggered, the UART receiver will behave in the same way as it does when the STOPRX task is triggered.

## 6.27.7 Error conditions

An ERROR event, in the form of a framing error, will be generated if a valid stop bit is not detected in a frame. Another ERROR event, in the form of a break condition, will be generated if the RXD line is held active low for longer than the length of a data frame. Effectively, a framing error is always generated before a break condition occurs.

## 6.27.8 Using the UART without flow control

If flow control is not enabled, the interface will behave as if the CTS and RTS lines are kept active all the time.

## 6.27.9 Parity and stop bit configuration

Automatic even parity generation for both transmission and reception can be configured using the register CONFIG on page 392. See the register description for details.

The amount of stop bits can also be configurated through the register CONFIG on page 392.

## 6.27.10 Registers

| Base address | Peripheral | Instance | Description | Configuration | |
|---|---|---|---|---|---|
| 0x40002000 | UART | UART0 | Universal asynchronous receiver/ transmitter | | Deprecated |

*Table 113: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_STARTRX | 0x000 | Start UART receiver |
| TASKS_STOPRX | 0x004 | Stop UART receiver |
| TASKS_STARTTX | 0x008 | Start UART transmitter |
| TASKS_STOPTX | 0x00C | Stop UART transmitter |
| TASKS_SUSPEND | 0x01C | Suspend UART |
| EVENTS_CTS | 0x100 | CTS is activated (set low). Clear To Send. |
| EVENTS_NCTS | 0x104 | CTS is deactivated (set high). Not Clear To Send. |
| EVENTS_RXDRDY | 0x108 | Data received in RXD |
| EVENTS_TXDRDY | 0x11C | Data sent from TXD |
| EVENTS_ERROR | 0x124 | Error detected |
| EVENTS_RXTO | 0x144 | Receiver timeout |
| SHORTS | 0x200 | Shortcuts between local events and tasks |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| ERRORSRC | 0x480 | Error source |
| ENABLE | 0x500 | Enable UART |
| PSEL.RTS | 0x508 | Pin select for RTS |
| PSEL.TXD | 0x50C | Pin select for TXD |
| PSEL.CTS | 0x510 | Pin select for CTS |
| PSEL.RXD | 0x514 | Pin select for RXD |
| RXD | 0x518 | RXD register |
| TXD | 0x51C | TXD register |
| BAUDRATE | 0x524 | Baud rate. Accuracy depends on the HFCLK source selected. |
| CONFIG | 0x56C | Configuration of parity and hardware flow control |

*Table 114: Register overview*

## 6.27.10.1 TASKS_STARTRX

Address offset: 0x000

Start UART receiver

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_STARTRX | | | Start UART receiver |
| | | | Trigger | 1 | Trigger task |

## 6.27.10.2 TASKS_STOPRX

Address offset: 0x004

Stop UART receiver

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_STOPRX | | | Stop UART receiver |
| | | | Trigger | 1 | Trigger task |

### 6.27.10.3 TASKS_STARTTX

Address offset: 0x008

Start UART transmitter

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_STARTTX | | | Start UART transmitter |
| | | | Trigger | 1 | Trigger task |

### 6.27.10.4 TASKS_STOPTX

Address offset: 0x00C

Stop UART transmitter

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_STOPTX | | | Stop UART transmitter |
| | | | Trigger | 1 | Trigger task |

### 6.27.10.5 TASKS_SUSPEND

Address offset: 0x01C

Suspend UART

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_SUSPEND | | | Suspend UART |
| | | | Trigger | 1 | Trigger task |

### 6.27.10.6 EVENTS_CTS

Address offset: 0x100

CTS is activated (set low). Clear To Send.

| Bit number | 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 |
|---|---|
| ID | A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | EVENTS_CTS | | | CTS is activated (set low). Clear To Send. |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.27.10.7 EVENTS_NCTS

Address offset: 0x104

CTS is deactivated (set high). Not Clear To Send.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_NCTS | | | CTS is deactivated (set high). Not Clear To Send. |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.27.10.8 EVENTS_RXDRDY

Address offset: 0x108

Data received in RXD

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_RXDRDY | | | Data received in RXD |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.27.10.9 EVENTS_TXDRDY

Address offset: 0x11C

Data sent from TXD

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_TXDRDY | | | Data sent from TXD |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

### 6.27.10.10 EVENTS_ERROR

Address offset: 0x124

Error detected

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_ERROR | | | Error detected |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.27.10.11 EVENTS_RXTO

Address offset: 0x144

Receiver timeout

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_RXTO | | | Receiver timeout |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.27.10.12 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | B A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | CTS_STARTRX | | | Shortcut between event CTS and task STARTRX |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| B | RW | NCTS_STOPRX | | | Shortcut between event NCTS and task STOPRX |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |

## 6.27.10.13 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | F E D C B A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | CTS | | | Write '1' to enable interrupt for event CTS |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | NCTS | | | Write '1' to enable interrupt for event NCTS |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | RXDRDY | | | Write '1' to enable interrupt for event RXDRDY |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | TXDRDY | | | Write '1' to enable interrupt for event TXDRDY |
| | | | Set | 1 | Enable |

| Bit number | 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 |
|---|---|
| ID | F E D C B A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E | RW | ERROR | | | Write '1' to enable interrupt for event ERROR |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| F | RW | RXTO | | | Write '1' to enable interrupt for event RXTO |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.27.10.14 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | 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 |
|---|---|
| ID | F E D C B A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | CTS | | | Write '1' to disable interrupt for event CTS |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | NCTS | | | Write '1' to disable interrupt for event NCTS |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | RXDRDY | | | Write '1' to disable interrupt for event RXDRDY |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | TXDRDY | | | Write '1' to disable interrupt for event TXDRDY |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E | RW | ERROR | | | Write '1' to disable interrupt for event ERROR |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| F | RW | RXTO | | | Write '1' to disable interrupt for event RXTO |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.27.10.15 ERRORSRC

Address offset: 0x480

Error source

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | OVERRUN | | | Overrun error<br>A start bit is received while the previous data still lies in RXD. (Previous data is lost.) |
| | | | NotPresent | 0 | Read: error not present |
| | | | Present | 1 | Read: error present |
| B | RW | PARITY | | | Parity error<br>A character with bad parity is received, if HW parity check is enabled. |
| | | | NotPresent | 0 | Read: error not present |
| | | | Present | 1 | Read: error present |
| C | RW | FRAMING | | | Framing error occurred<br>A valid stop bit is not detected on the serial data input after all bits in a character have been received. |
| | | | NotPresent | 0 | Read: error not present |
| | | | Present | 1 | Read: error present |
| D | RW | BREAK | | | Break condition<br>The serial data input is '0' for longer than the length of a data frame. (The data frame length is 10 bits without parity bit, and 11 bits with parity bit.). |
| | | | NotPresent | 0 | Read: error not present |
| | | | Present | 1 | Read: error present |

## 6.27.10.16 ENABLE

Address offset: 0x500

Enable UART

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ENABLE | | | Enable or disable UART |
| | | | Disabled | 0 | Disable UART |
| | | | Enabled | 4 | Enable UART |

## 6.27.10.17 PSEL.RTS

Address offset: 0x508

Pin select for RTS

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.27.10.18 PSEL.TXD

Address offset: 0x50C

Pin select for TXD

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.27.10.19 PSEL.CTS

Address offset: 0x510

Pin select for CTS

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.27.10.20 PSEL.RXD

Address offset: 0x514

Pin select for RXD

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.27.10.21 RXD

Address offset: 0x518

RXD register

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | RXD | | | RX data received in previous transfers, double buffered |

## 6.27.10.22 TXD

Address offset: 0x51C

TXD register

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TXD | | | TX data to be transferred |

## 6.27.10.23 BAUDRATE

Address offset: 0x524

Baud rate. Accuracy depends on the HFCLK source selected.

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x04000000** | **0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | BAUDRATE | | | Baud rate |
| | | | Baud1200 | 0x0004F000 | 1200 baud (actual rate: 1205) |
| | | | Baud2400 | 0x0009D000 | 2400 baud (actual rate: 2396) |
| | | | Baud4800 | 0x0013B000 | 4800 baud (actual rate: 4808) |
| | | | Baud9600 | 0x00275000 | 9600 baud (actual rate: 9598) |
| | | | Baud14400 | 0x003B0000 | 14400 baud (actual rate: 14414) |
| | | | Baud19200 | 0x004EA000 | 19200 baud (actual rate: 19208) |
| | | | Baud28800 | 0x0075F000 | 28800 baud (actual rate: 28829) |
| | | | Baud31250 | 0x00800000 | 31250 baud |
| | | | Baud38400 | 0x009D5000 | 38400 baud (actual rate: 38462) |
| | | | Baud56000 | 0x00E50000 | 56000 baud (actual rate: 55944) |
| | | | Baud57600 | 0x00EBF000 | 57600 baud (actual rate: 57762) |
| | | | Baud76800 | 0x013A9000 | 76800 baud (actual rate: 76923) |
| | | | Baud115200 | 0x01D7E000 | 115200 baud (actual rate: 115942) |
| | | | Baud230400 | 0x03AFB000 | 230400 baud (actual rate: 231884) |
| | | | Baud250000 | 0x04000000 | 250000 baud |
| | | | Baud460800 | 0x075F7000 | 460800 baud (actual rate: 470588) |
| | | | Baud921600 | 0x0EBED000 | 921600 baud (actual rate: 941176) |
| | | | Baud1M | 0x10000000 | 1Mega baud |

### 6.27.10.24 CONFIG

Address offset: 0x56C

Configuration of parity and hardware flow control

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | C B B B A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | HWFC | | | Hardware flow control |
| | | | Disabled | 0 | Disabled |
| | | | Enabled | 1 | Enabled |
| B | RW | PARITY | | | Parity |
| | | | Excluded | 0x0 | Exclude parity bit |
| | | | Included | 0x7 | Include parity bit |
| C | RW | STOP | | | Stop bits |
| | | | One | 0 | One stop bit |
| | | | Two | 1 | Two stop bits |

## 6.27.11 Electrical specification

### 6.27.11.1 UART electrical specification

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $f_{UART}$ | Baud rate for UART[32]. | | | 1000 | kbps |
| $t_{UART,CTSH}$ | CTS high time | 1 | | | µs |
| $t_{UART,START}$ | Time from STARTRX/STARTTX task to transmission started | | 1 | | µs |

# 6.28 UARTE — Universal asynchronous receiver/transmitter with EasyDMA

The Universal asynchronous receiver/transmitter with EasyDMA (UARTE) offers fast, full-duplex, asynchronous serial communication with built-in flow control (CTS, RTS) support in hardware at a rate up to 1 Mbps, and EasyDMA data transfer from/to RAM.

Listed here are the main features for UARTE:

- Full-duplex operation
- Automatic hardware flow control
- Optional even parity bit checking and generation
- EasyDMA
- Up to 1 Mbps baudrate
- Return to IDLE between transactions supported (when using HW flow control)
- One or two stop bit
- Least significant bit (LSB) first

[32] High baud rates may require GPIOs to be set as High Drive, see GPIO chapter for more details.

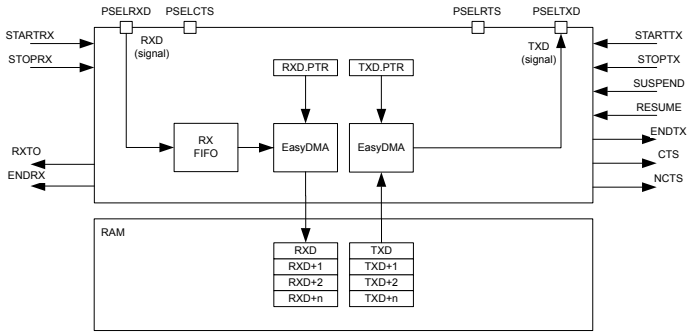



*Figure 147: UARTE configuration*

The GPIOs used for each UART interface can be chosen from any GPIO on the device and are independently configurable. This enables great flexibility in device pinout and efficient use of board space and signal routing.

**Note:** External crystal oscillator must be enabled to obtain sufficient clock accuracy for stable communication. See CLOCK — Clock control on page 61 for more information.

## 6.28.1 EasyDMA

The UARTE implements EasyDMA for reading and writing to and from the RAM.

If the TXD.PTR and the RXD.PTR are not pointing to the Data RAM region, an EasyDMA transfer may result in a HardFault or RAM corruption. See Memory on page 16 for more information about the different memory regions.

The .PTR and .MAXCNT registers are double-buffered. They can be updated and prepared for the next RX/TX transmission immediately after having received the RXSTARTED/TXSTARTED event.

The ENDRX/ENDTX event indicates that EasyDMA has finished accessing respectively the RX/TX buffer in RAM.

## 6.28.2 Transmission

The first step of a DMA transmission is storing bytes in the transmit buffer and configuring EasyDMA. This is achieved by writing the initial address pointer to TXD.PTR, and the number of bytes in the RAM buffer to TXD.MAXCNT. The UARTE transmission is started by triggering the STARTTX task.

After each byte has been sent over the TXD line, a TXDRDY event will be generated.

When all bytes in the TXD buffer, as specified in the TXD.MAXCNT register, have been transmitted, the UARTE transmission will end automatically and an ENDTX event will be generated.

A UARTE transmission sequence is stopped by triggering the STOPTX task, a TXSTOPPED event will be generated when the UARTE transmitter has stopped.

If the ENDTX event has not already been generated when the UARTE transmitter has come to a stop, the UARTE will generate the ENDTX event explicitly even though all bytes in the TXD buffer, as specified in the TXD.MAXCNT register, have not been transmitted.

If flow control is enabled through the HWFC field in the CONFIG register, a transmission will be automatically suspended when CTS is deactivated and resumed when CTS is activated again, as illustrated in UARTE transmission on page 394. A byte that is in transmission when CTS is deactivated will be fully transmitted before the transmission is suspended.

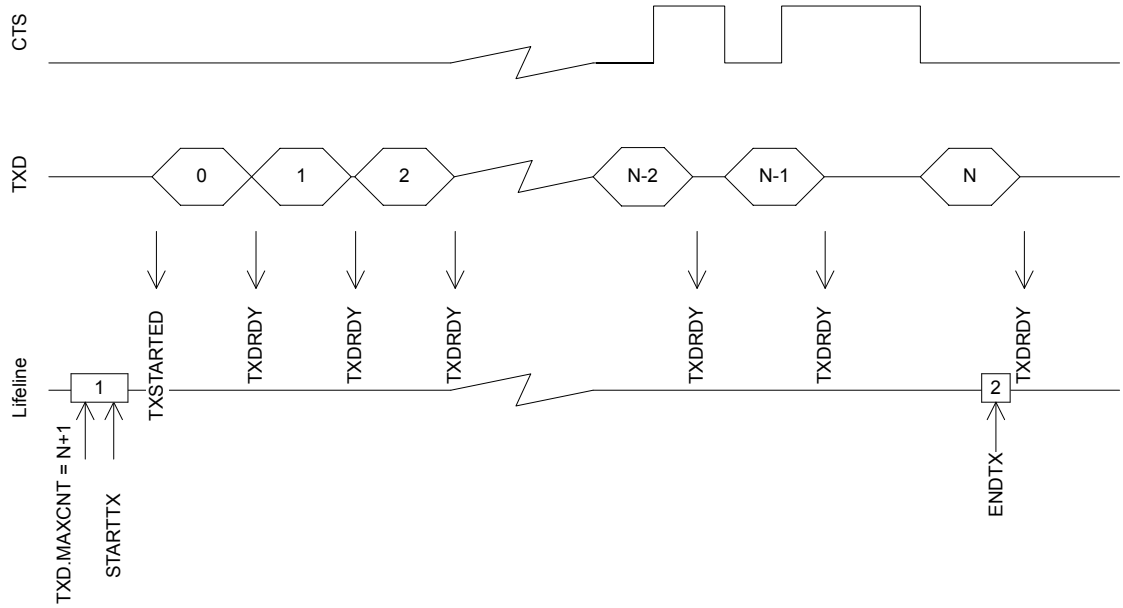



*Figure 148: UARTE transmission*

The UARTE transmitter will be in its lowest activity level, and consume the least amount of energy, when it is stopped, i.e. before it is started via STARTTX or after it has been stopped via STOPTX and the TXSTOPPED event has been generated. See POWER — Power supply on page 48 for more information about power modes.

## 6.28.3 Reception

The UARTE receiver is started by triggering the STARTRX task. The UARTE receiver is using EasyDMA to store incoming data in an RX buffer in RAM.

The RX buffer is located at the address specified in the RXD.PTR register. The RXD.PTR register is double-buffered and it can be updated and prepared for the next STARTRX task immediately after the RXSTARTED event is generated. The size of the RX buffer is specified in the RXD.MAXCNT register and the UARTE will generate an ENDRX event when it has filled up the RX buffer, see UARTE reception on page 395.

For each byte received over the RXD line, an RXDRDY event will be generated. This event is likely to occur before the corresponding data has been transferred to Data RAM.

The RXD.AMOUNT register can be queried following an ENDRX event to see how many new bytes have been transferred to the RX buffer in RAM since the previous ENDRX event.

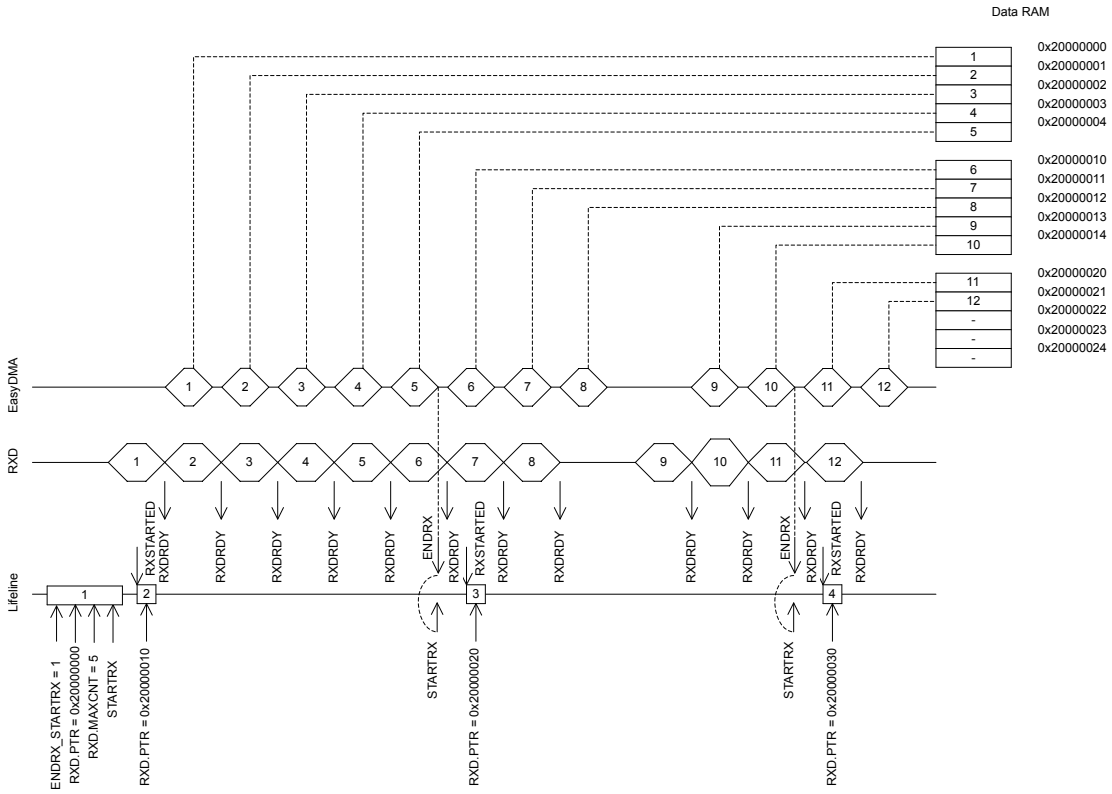



*Figure 149: UARTE reception*

The UARTE receiver is stopped by triggering the STOPRX task. An RXTO event is generated when the UARTE has stopped. The UARTE will make sure that an impending ENDRX event will be generated before the RXTO event is generated. This means that the UARTE will guarantee that no ENDRX event will be generated after RXTO, unless the UARTE is restarted or a FLUSHRX command is issued after the RXTO event is generated.

**Important:** If the ENDRX event has not already been generated when the UARTE receiver has come to a stop, which implies that all pending content in the RX FIFO has been moved to the RX buffer, the UARTE will generate the ENDRX event explicitly even though the RX buffer is not full. In this scenario the ENDRX event will be generated before the RXTO event is generated.

To be able to know how many bytes have actually been received into the RX buffer, the CPU can read the RXD.AMOUNT register following the ENDRX event or the RXTO event.

The UARTE is able to receive up to four bytes after the STOPRX task has been triggered as long as these are sent in succession immediately after the RTS signal is deactivated. This is possible because after the RTS is deactivated the UARTE is able to receive bytes for an extended period equal to the time it takes to send 4 bytes on the configured baud rate.

After the RXTO event is generated the internal RX FIFO may still contain data, and to move this data to RAM the FLUSHRX task must be triggered. To make sure that this data does not overwrite data in the RX buffer, the RX buffer should be emptied or the RXD.PTR should be updated before the FLUSHRX task is triggered. To make sure that all data in the RX FIFO is moved to the RX buffer, the RXD.MAXCNT register must be set to RXD.MAXCNT > 4, see UARTE reception with forced stop via STOPRX on page 396. The UARTE will generate the ENDRX event after completing the FLUSHRX task even if the RX FIFO was empty or if the RX buffer does not get filled up. To be able to know how many bytes have actually been received into the RX buffer in this case, the CPU can read the RXD.AMOUNT register following the ENDRX event.

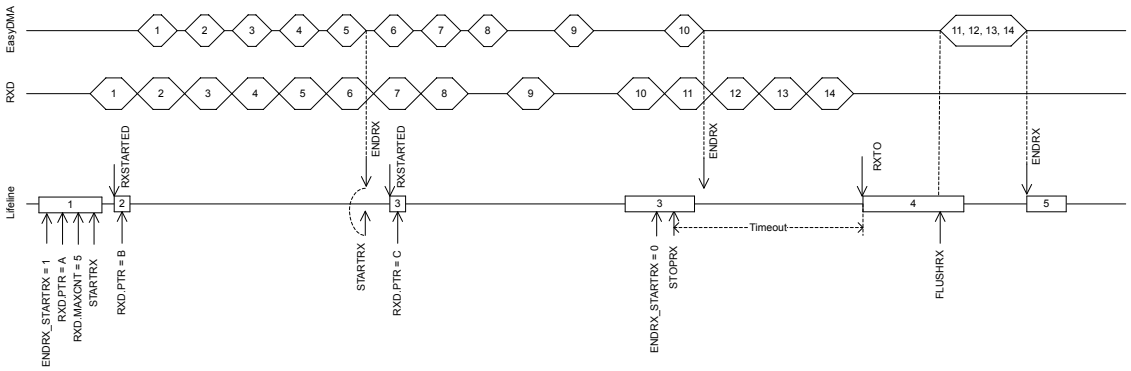



*Figure 150: UARTE reception with forced stop via STOPRX*

If HW flow control is enabled through the HWFC field in the CONFIG register, the RTS signal will be deactivated when the receiver is stopped via the STOPRX task or when the UARTE is only able to receive four more bytes in its internal RX FIFO.

With flow control disabled, the UARTE will function in the same way as when the flow control is enabled except that the RTS line will not be used. This means that no signal will be generated when the UARTE has reached the point where it is only able to receive four more bytes in its internal RX FIFO. Data received when the internal RX FIFO is filled up, will be lost.

The UARTE receiver will be in its lowest activity level, and consume the least amount of energy, when it is stopped, i.e. before it is started via STARTRX or after it has been stopped via STOPRX and the RXTO event has been generated. See POWER — Power supply on page 48 for more information about power modes.

## 6.28.4 Error conditions

An ERROR event, in the form of a framing error, will be generated if a valid stop bit is not detected in a frame. Another ERROR event, in the form of a break condition, will be generated if the RXD line is held active low for longer than the length of a data frame. Effectively, a framing error is always generated before a break condition occurs.

An ERROR event will not stop reception. If the error was a parity error, the received byte will still be transferred into Data RAM, and so will following incoming bytes. If there was a framing error (wrong stop bit), that specific byte will NOT be stored into Data RAM, but following incoming bytes will.

## 6.28.5 Using the UARTE without flow control

If flow control is not enabled, the interface will behave as if the CTS and RTS lines are kept active all the time.

## 6.28.6 Parity and stop bit configuration

Automatic even parity generation for both transmission and reception can be configured using the register CONFIG on page 410. See the register description for details.

The amount of stop bits can also be configured through the register CONFIG on page 410.

## 6.28.7 Low power

When putting the system in low power and the peripheral is not needed, lowest possible power consumption is achieved by stopping, and then disabling the peripheral.

The STOPTX and STOPRX tasks may not be always needed (the peripheral might already be stopped), but if STOPTX and/or STOPRX is sent, software shall wait until the TXSTOPPED and/or RXTO event is received in response, before disabling the peripheral through the ENABLE register.

## 6.28.8 Pin configuration

The different signals RXD, CTS (Clear To Send, active low), RTS (Request To Send, active low), and TXD associated with the UARTE are mapped to physical pins according to the configuration specified in the PSEL.RXD, PSEL.CTS, PSEL.RTS, and PSEL.TXD registers respectively.

The PSEL.RXD, PSEL.CTS, PSEL.RTS, and PSEL.TXD registers and their configurations are only used as long as the UARTE is enabled, and retained only for the duration the device is in ON mode. PSEL.RXD, PSEL.RTS, PSEL.RTS and PSEL.TXD must only be configured when the UARTE is disabled.

To secure correct signal levels on the pins by the UARTE when the system is in OFF mode, the pins must be configured in the GPIO peripheral as described in GPIO configuration before enabling peripheral on page 397.

Only one peripheral can be assigned to drive a particular GPIO pin at a time. Failing to do so may result in unpredictable behavior.

| UARTE signal | UARTE pin | Direction | Output value |
|---|---|---|---|
| RXD | As specified in PSEL.RXD | Input | Not applicable |
| CTS | As specified in PSEL.CTS | Input | Not applicable |
| RTS | As specified in PSEL.RTS | Output | 1 |
| TXD | As specified in PSEL.TXD | Output | 1 |

*Table 115: GPIO configuration before enabling peripheral*

## 6.28.9 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40002000 | UARTE | UARTE0 | Universal asynchronous receiver/ transmitter with EasyDMA | |

*Table 116: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_STARTRX | 0x000 | Start UART receiver |
| TASKS_STOPRX | 0x004 | Stop UART receiver |
| TASKS_STARTTX | 0x008 | Start UART transmitter |
| TASKS_STOPTX | 0x00C | Stop UART transmitter |
| TASKS_FLUSHRX | 0x02C | Flush RX FIFO into RX buffer |
| EVENTS_CTS | 0x100 | CTS is activated (set low). Clear To Send. |
| EVENTS_NCTS | 0x104 | CTS is deactivated (set high). Not Clear To Send. |
| EVENTS_RXDRDY | 0x108 | Data received in RXD (but potentially not yet transferred to Data RAM) |
| EVENTS_ENDRX | 0x110 | Receive buffer is filled up |
| EVENTS_TXDRDY | 0x11C | Data sent from TXD |
| EVENTS_ENDTX | 0x120 | Last TX byte transmitted |
| EVENTS_ERROR | 0x124 | Error detected |
| EVENTS_RXTO | 0x144 | Receiver timeout |
| EVENTS_RXSTARTED | 0x14C | UART receiver has started |
| EVENTS_TXSTARTED | 0x150 | UART transmitter has started |
| EVENTS_TXSTOPPED | 0x158 | Transmitter stopped |
| SHORTS | 0x200 | Shortcuts between local events and tasks |
| INTEN | 0x300 | Enable or disable interrupt |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |

| Register | Offset | Description |
|---|---|---|
| ERRORSRC | 0x480 | Error source<br><br>Note : this register is read / write one to clear. |
| ENABLE | 0x500 | Enable UART |
| PSEL.RTS | 0x508 | Pin select for RTS signal |
| PSEL.TXD | 0x50C | Pin select for TXD signal |
| PSEL.CTS | 0x510 | Pin select for CTS signal |
| PSEL.RXD | 0x514 | Pin select for RXD signal |
| BAUDRATE | 0x524 | Baud rate. Accuracy depends on the HFCLK source selected. |
| RXD.PTR | 0x534 | Data pointer |
| RXD.MAXCNT | 0x538 | Maximum number of bytes in receive buffer |
| RXD.AMOUNT | 0x53C | Number of bytes transferred in the last transaction |
| TXD.PTR | 0x544 | Data pointer |
| TXD.MAXCNT | 0x548 | Maximum number of bytes in transmit buffer |
| TXD.AMOUNT | 0x54C | Number of bytes transferred in the last transaction |
| CONFIG | 0x56C | Configuration of parity and hardware flow control |

*Table 117: Register overview*

## 6.28.9.1 TASKS_STARTRX

Address offset: 0x000

Start UART receiver

| Bit number | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | A |
| **Reset 0x00000000** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_STARTRX | | | Start UART receiver |
| | | | Trigger | 1 | Trigger task |

## 6.28.9.2 TASKS_STOPRX

Address offset: 0x004

Stop UART receiver

| Bit number | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | A |
| **Reset 0x00000000** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | W | TASKS_STOPRX | | | Stop UART receiver |
| | | | Trigger | 1 | Trigger task |

## 6.28.9.3 TASKS_STARTTX

Address offset: 0x008

Start UART transmitter

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STARTTX | | | Start UART transmitter |
| | | | Trigger | 1 | Trigger task |

## 6.28.9.4 TASKS_STOPTX

Address offset: 0x00C

Stop UART transmitter

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_STOPTX | | | Stop UART transmitter |
| | | | Trigger | 1 | Trigger task |

## 6.28.9.5 TASKS_FLUSHRX

Address offset: 0x02C

Flush RX FIFO into RX buffer

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_FLUSHRX | | | Flush RX FIFO into RX buffer |
| | | | Trigger | 1 | Trigger task |

## 6.28.9.6 EVENTS_CTS

Address offset: 0x100

CTS is activated (set low). Clear To Send.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_CTS | | | CTS is activated (set low). Clear To Send. |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.28.9.7 EVENTS_NCTS

Address offset: 0x104

CTS is deactivated (set high). Not Clear To Send.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_NCTS | | | CTS is deactivated (set high). Not Clear To Send. |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.28.9.8 EVENTS_RXDRDY

Address offset: 0x108

Data received in RXD (but potentially not yet transferred to Data RAM)

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_RXDRDY | | | Data received in RXD (but potentially not yet transferred to Data RAM) |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.28.9.9 EVENTS_ENDRX

Address offset: 0x110

Receive buffer is filled up

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_ENDRX | | | Receive buffer is filled up |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.28.9.10 EVENTS_TXDRDY

Address offset: 0x11C

Data sent from TXD

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_TXDRDY | | | Data sent from TXD |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.28.9.11 EVENTS_ENDTX

Address offset: 0x120

Last TX byte transmitted

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_ENDTX | | | Last TX byte transmitted |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.28.9.12 EVENTS_ERROR

Address offset: 0x124

Error detected

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_ERROR | | | Error detected |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.28.9.13 EVENTS_RXTO

Address offset: 0x144

Receiver timeout

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_RXTO | | | Receiver timeout |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.28.9.14 EVENTS_RXSTARTED

Address offset: 0x14C

UART receiver has started

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_RXSTARTED | | | UART receiver has started |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.28.9.15 EVENTS_TXSTARTED

Address offset: 0x150

UART transmitter has started

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_TXSTARTED | | | UART transmitter has started |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.28.9.16 EVENTS_TXSTOPPED

Address offset: 0x158

Transmitter stopped

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_TXSTOPPED | | | Transmitter stopped |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.28.9.17 SHORTS

Address offset: 0x200

Shortcuts between local events and tasks

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | D C |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| C | RW | ENDRX_STARTRX | | | Shortcut between event ENDRX and task STARTRX |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |
| D | RW | ENDRX_STOPRX | | | Shortcut between event ENDRX and task STOPRX |
| | | | Disabled | 0 | Disable shortcut |
| | | | Enabled | 1 | Enable shortcut |

## 6.28.9.18 INTEN

Address offset: 0x300

Enable or disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | L J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | CTS | | | Enable or disable interrupt for event CTS |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| B | RW | NCTS | | | Enable or disable interrupt for event NCTS |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | L J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| C | RW | RXDRDY | | | Enable or disable interrupt for event RXDRDY |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| D | RW | ENDRX | | | Enable or disable interrupt for event ENDRX |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| E | RW | TXDRDY | | | Enable or disable interrupt for event TXDRDY |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| F | RW | ENDTX | | | Enable or disable interrupt for event ENDTX |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| G | RW | ERROR | | | Enable or disable interrupt for event ERROR |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| H | RW | RXTO | | | Enable or disable interrupt for event RXTO |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| I | RW | RXSTARTED | | | Enable or disable interrupt for event RXSTARTED |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| J | RW | TXSTARTED | | | Enable or disable interrupt for event TXSTARTED |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |
| L | RW | TXSTOPPED | | | Enable or disable interrupt for event TXSTOPPED |
| | | | Disabled | 0 | Disable |
| | | | Enabled | 1 | Enable |

## 6.28.9.19 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | L J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | CTS | | | Write '1' to enable interrupt for event CTS |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | NCTS | | | Write '1' to enable interrupt for event NCTS |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | RXDRDY | | | Write '1' to enable interrupt for event RXDRDY |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | L J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| D | RW | ENDRX | | | Write '1' to enable interrupt for event ENDRX |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E | RW | TXDRDY | | | Write '1' to enable interrupt for event TXDRDY |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| F | RW | ENDTX | | | Write '1' to enable interrupt for event ENDTX |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| G | RW | ERROR | | | Write '1' to enable interrupt for event ERROR |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| H | RW | RXTO | | | Write '1' to enable interrupt for event RXTO |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| I | RW | RXSTARTED | | | Write '1' to enable interrupt for event RXSTARTED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| J | RW | TXSTARTED | | | Write '1' to enable interrupt for event TXSTARTED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| L | RW | TXSTOPPED | | | Write '1' to enable interrupt for event TXSTOPPED |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.28.9.20 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | L J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | CTS | | | Write '1' to disable interrupt for event CTS |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| B | RW | NCTS | | | Write '1' to disable interrupt for event NCTS |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | L J I H G F E D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| | | | Enabled | 1 | Read: Enabled |
| C | RW | RXDRDY | | | Write '1' to disable interrupt for event RXDRDY |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| D | RW | ENDRX | | | Write '1' to disable interrupt for event ENDRX |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| E | RW | TXDRDY | | | Write '1' to disable interrupt for event TXDRDY |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| F | RW | ENDTX | | | Write '1' to disable interrupt for event ENDTX |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| G | RW | ERROR | | | Write '1' to disable interrupt for event ERROR |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| H | RW | RXTO | | | Write '1' to disable interrupt for event RXTO |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| I | RW | RXSTARTED | | | Write '1' to disable interrupt for event RXSTARTED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| J | RW | TXSTARTED | | | Write '1' to disable interrupt for event TXSTARTED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |
| L | RW | TXSTOPPED | | | Write '1' to disable interrupt for event TXSTOPPED |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

## 6.28.9.21 ERRORSRC

Address offset: 0x480

Error source

Note : this register is read / write one to clear.

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | D C B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | OVERRUN | | | Overrun error<br>A start bit is received while the previous data still lies in RXD. (Previous data is lost.) |
| | | | NotPresent | 0 | Read: error not present |
| | | | Present | 1 | Read: error present |
| B | RW | PARITY | | | Parity error<br>A character with bad parity is received, if HW parity check is enabled. |
| | | | NotPresent | 0 | Read: error not present |
| | | | Present | 1 | Read: error present |
| C | RW | FRAMING | | | Framing error occurred<br>A valid stop bit is not detected on the serial data input after all bits in a character have been received. |
| | | | NotPresent | 0 | Read: error not present |
| | | | Present | 1 | Read: error present |
| D | RW | BREAK | | | Break condition<br>The serial data input is '0' for longer than the length of a data frame. (The data frame length is 10 bits without parity bit, and 11 bits with parity bit.). |
| | | | NotPresent | 0 | Read: error not present |
| | | | Present | 1 | Read: error present |

## 6.28.9.22 ENABLE

Address offset: 0x500

Enable UART

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | ENABLE | | | Enable or disable UARTE |
| | | | Disabled | 0 | Disable UARTE |
| | | | Enabled | 8 | Enable UARTE |

## 6.28.9.23 PSEL.RTS

Address offset: 0x508

Pin select for RTS signal

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.28.9.24 PSEL.TXD

Address offset: 0x50C

Pin select for TXD signal

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.28.9.25 PSEL.CTS

Address offset: 0x510

Pin select for CTS signal

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.28.9.26 PSEL.RXD

Address offset: 0x514

Pin select for RXD signal

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PIN | | [0..31] | Pin number |
| C | RW | CONNECT | | | Connection |
| | | | Disconnected | 1 | Disconnect |
| | | | Connected | 0 | Connect |

## 6.28.9.27 BAUDRATE

Address offset: 0x524

Baud rate. Accuracy depends on the HFCLK source selected.

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x04000000** | | | **0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | BAUDRATE | | | Baud rate |
| | | | Baud1200 | 0x0004F000 | 1200 baud (actual rate: 1205) |
| | | | Baud2400 | 0x0009D000 | 2400 baud (actual rate: 2396) |
| | | | Baud4800 | 0x0013B000 | 4800 baud (actual rate: 4808) |
| | | | Baud9600 | 0x00275000 | 9600 baud (actual rate: 9598) |
| | | | Baud14400 | 0x003AF000 | 14400 baud (actual rate: 14401) |
| | | | Baud19200 | 0x004EA000 | 19200 baud (actual rate: 19208) |
| | | | Baud28800 | 0x0075C000 | 28800 baud (actual rate: 28777) |
| | | | Baud31250 | 0x00800000 | 31250 baud |
| | | | Baud38400 | 0x009D0000 | 38400 baud (actual rate: 38369) |
| | | | Baud56000 | 0x00E50000 | 56000 baud (actual rate: 55944) |
| | | | Baud57600 | 0x00EB0000 | 57600 baud (actual rate: 57554) |
| | | | Baud76800 | 0x013A9000 | 76800 baud (actual rate: 76923) |
| | | | Baud115200 | 0x01D60000 | 115200 baud (actual rate: 115108) |
| | | | Baud230400 | 0x03B00000 | 230400 baud (actual rate: 231884) |
| | | | Baud250000 | 0x04000000 | 250000 baud |
| | | | Baud460800 | 0x07400000 | 460800 baud (actual rate: 457143) |
| | | | Baud921600 | 0x0F000000 | 921600 baud (actual rate: 941176) |
| | | | Baud1M | 0x10000000 | 1Mega baud |

## 6.28.9.28 RXD.PTR

Address offset: 0x534

Data pointer

| Bit number | | | 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 |
|---|---|---|---|
| ID | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | PTR | | | Data pointer |

**Note:** See the memory chapter for details about which memories are available for EasyDMA.

## 6.28.9.29 RXD.MAXCNT

Address offset: 0x538

Maximum number of bytes in receive buffer

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | MAXCNT | | [1..0x3FF] | Maximum number of bytes in receive buffer |

## 6.28.9.30 RXD.AMOUNT

Address offset: 0x53C

Number of bytes transferred in the last transaction

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | R | AMOUNT | | [1..0x3FF] | Number of bytes transferred in the last transaction |

## 6.28.9.31 TXD.PTR

Address offset: 0x544

Data pointer

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | PTR | | | Data pointer |

**Note:** See the memory chapter for details about which memories are available for EasyDMA.

## 6.28.9.32 TXD.MAXCNT

Address offset: 0x548

Maximum number of bytes in transmit buffer

| Bit number | 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 |
|---|---|
| ID | A A A A A A A A A |
| **Reset 0x00000000** | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |

| ID | Acce | Field | Value ID | Value | Description |
|---|---|---|---|---|---|
| A | RW | MAXCNT | | [1..0x3FF] | Maximum number of bytes in transmit buffer |

## 6.28.9.33 TXD.AMOUNT

Address offset: 0x54C

Number of bytes transferred in the last transaction

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | AMOUNT | | [1..0x3FF] | Number of bytes transferred in the last transaction |

### 6.28.9.34 CONFIG

Address offset: 0x56C

Configuration of parity and hardware flow control

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C B B B A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | HWFC | | | Hardware flow control |
| | | | Disabled | 0 | Disabled |
| | | | Enabled | 1 | Enabled |
| B | RW | PARITY | | | Parity |
| | | | Excluded | 0x0 | Exclude parity bit |
| | | | Included | 0x7 | Include even parity bit |
| C | RW | STOP | | | Stop bits |
| | | | One | 0 | One stop bit |
| | | | Two | 1 | Two stop bits |

## 6.28.10 Electrical specification

### 6.28.10.1 UARTE electrical specification

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $f_{UARTE}$ | Baud rate for UARTE[33]. | | | 1000 | kbps |
| $t_{UARTE,CTSH}$ | CTS high time | 1 | | | µs |
| $t_{UARTE,START}$ | Time from STARTRX/STARTTX task to transmission started | .. | .. | .. | µs |

# 6.29 WDT — Watchdog timer

A countdown watchdog timer using the low-frequency clock source (LFCLK) offers configurable and robust protection against application lock-up.

The watchdog timer is started by triggering the START task.

The watchdog can be paused during long CPU sleep periods for low power applications and when the debugger has halted the CPU. The watchdog is implemented as a down-counter that generates a TIMEOUT event when it wraps over after counting down to 0. When the watchdog timer is started through the START task, the watchdog counter is loaded with the value specified in the CRV register. This counter is also reloaded with the value specified in the CRV register when a reload request is granted.

[33] High baud rates may require GPIOs to be set as High Drive, see GPIO chapter for more details.

The watchdog's timeout period is given by:

```
timeout [s] = ( CRV + 1 ) / 32768
```

When started, the watchdog will automatically force the 32.768 kHz RC oscillator on as long as no other 32.768 kHz clock source is running and generating the 32.768 kHz system clock, see chapter CLOCK — Clock control on page 61.

## 6.29.1 Reload criteria

The watchdog has eight separate reload request registers, which shall be used to request the watchdog to reload its counter with the value specified in the CRV register. To reload the watchdog counter, the special value 0x6E524635 needs to be written to all enabled reload registers.

One or more RR registers can be individually enabled through the RREN register.

## 6.29.2 Temporarily pausing the watchdog

By default, the watchdog will be active counting down the down-counter while the CPU is sleeping and when it is halted by the debugger. It is however possible to configure the watchdog to automatically pause while the CPU is sleeping as well as when it is halted by the debugger.

## 6.29.3 Watchdog reset

A TIMEOUT event will automatically lead to a watchdog reset.

See Reset on page 52 for more information about reset sources. If the watchdog is configured to generate an interrupt on the TIMEOUT event, the watchdog reset will be postponed with two 32.768 kHz clock cycles after the TIMEOUT event has been generated. Once the TIMEOUT event has been generated, the impending watchdog reset will always be effectuated.

The watchdog must be configured before it is started. After it is started, the watchdog's configuration registers, which comprise registers CRV, RREN, and CONFIG, will be blocked for further configuration.

The watchdog can be reset from several reset sources, see Reset behavior on page 53.

When the device starts running again, after a reset, or waking up from OFF mode, the watchdog configuration registers will be available for configuration again.

## 6.29.4 Registers

| Base address | Peripheral | Instance | Description | Configuration |
|---|---|---|---|---|
| 0x40010000 | WDT | WDT | Watchdog timer | |

*Table 118: Instances*

| Register | Offset | Description |
|---|---|---|
| TASKS_START | 0x000 | Start the watchdog |
| EVENTS_TIMEOUT | 0x100 | Watchdog timeout |
| INTENSET | 0x304 | Enable interrupt |
| INTENCLR | 0x308 | Disable interrupt |
| RUNSTATUS | 0x400 | Run status |
| REQSTATUS | 0x404 | Request status |
| CRV | 0x504 | Counter reload value |
| RREN | 0x508 | Enable register for reload request registers |

| Register | Offset | Description |
|---|---|---|
| CONFIG | 0x50C | Configuration register |
| RR[0] | 0x600 | Reload request 0 |
| RR[1] | 0x604 | Reload request 1 |
| RR[2] | 0x608 | Reload request 2 |
| RR[3] | 0x60C | Reload request 3 |
| RR[4] | 0x610 | Reload request 4 |
| RR[5] | 0x614 | Reload request 5 |
| RR[6] | 0x618 | Reload request 6 |
| RR[7] | 0x61C | Reload request 7 |

*Table 119: Register overview*

## 6.29.4.1 TASKS_START

Address offset: 0x000

Start the watchdog

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | TASKS_START | | | Start the watchdog |
| | | | Trigger | 1 | Trigger task |

## 6.29.4.2 EVENTS_TIMEOUT

Address offset: 0x100

Watchdog timeout

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | EVENTS_TIMEOUT | | | Watchdog timeout |
| | | | NotGenerated | 0 | Event not generated |
| | | | Generated | 1 | Event generated |

## 6.29.4.3 INTENSET

Address offset: 0x304

Enable interrupt

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | TIMEOUT | | | Write '1' to enable interrupt for event TIMEOUT |
| | | | Set | 1 | Enable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

### 6.29.4.4 INTENCLR

Address offset: 0x308

Disable interrupt

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | TIMEOUT | | | Write '1' to disable interrupt for event TIMEOUT |
| | | | Clear | 1 | Disable |
| | | | Disabled | 0 | Read: Disabled |
| | | | Enabled | 1 | Read: Enabled |

### 6.29.4.5 RUNSTATUS

Address offset: 0x400

Run status

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | A |
| **Reset 0x00000000** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | R | RUNSTATUS | | | Indicates whether or not the watchdog is running |
| | | | NotRunning | 0 | Watchdog not running |
| | | | Running | 1 | Watchdog is running |

### 6.29.4.6 REQSTATUS

Address offset: 0x404

Request status

| Bit number | | | | | 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 |
|---|---|---|---|---|---|
| ID | | | | | H G F E D C B A |
| **Reset 0x00000001** | | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A-H | R | RR[i] (i=0..7) | | | Request status for RR[i] register |
| | | | DisabledOrRequested | 0 | RR[i] register is not enabled, or are already requesting reload |
| | | | EnabledAndUnrequested | 1 | RR[i] register is enabled, and are not yet requesting reload |

### 6.29.4.7 CRV

Address offset: 0x504

Counter reload value

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0xFFFFFFFF** | | | | **1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | CRV | | [0x0000000F..0xFFFFFFFF] | Counter reload value in number of cycles of the 32.768 kHz clock |

## 6.29.4.8 RREN

Address offset: 0x508

Enable register for reload request registers

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | H G F E D C B A |
| **Reset 0x00000001** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A-H | RW | RR[i] (i=0..7) | | | Enable or disable RR[i] register |
| | | | Disabled | 0 | Disable RR[i] register |
| | | | Enabled | 1 | Enable RR[i] register |

## 6.29.4.9 CONFIG

Address offset: 0x50C

Configuration register

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | C A |
| **Reset 0x00000001** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1** |
| ID | Acce | Field | Value ID | Value | Description |
| A | RW | SLEEP | | | Configure the watchdog to either be paused, or kept running, while the CPU is sleeping |
| | | | Pause | 0 | Pause watchdog while the CPU is sleeping |
| | | | Run | 1 | Keep the watchdog running while the CPU is sleeping |
| C | RW | HALT | | | Configure the watchdog to either be paused, or kept running, while the CPU is halted by the debugger |
| | | | Pause | 0 | Pause watchdog while the CPU is halted by the debugger |
| | | | Run | 1 | Keep the watchdog running while the CPU is halted by the debugger |

## 6.29.4.10 RR[n] (n=0..7)

Address offset: 0x600 + (n × 0x4)

Reload request n

| Bit number | | | | 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 |
|---|---|---|---|---|
| ID | | | | A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A |
| **Reset 0x00000000** | | | | **0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0** |
| ID | Acce | Field | Value ID | Value | Description |
| A | W | RR | | | Reload request register |
| | | | Reload | 0x6E524635 | Value to request a reload of the watchdog timer |

## 6.29.5 Electrical specification

### 6.29.5.1 Watchdog Timer Electrical Specification

| Symbol | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{WDT}$ | Time out interval | 458 µs | | 36 h | |

# 7 Hardware and layout

## 7.1 Pin assignments

The pin assignment figures and tables describe the pinouts for the product variants of the chip. There are also recommendations for how the GPIO pins should be configured, in addition to any usage restrictions.

### 7.1.1 QFN48 pin assignments

The nRF52811 QFN48 pin assignment table and figure describe the pinouts for this variant of the chip.

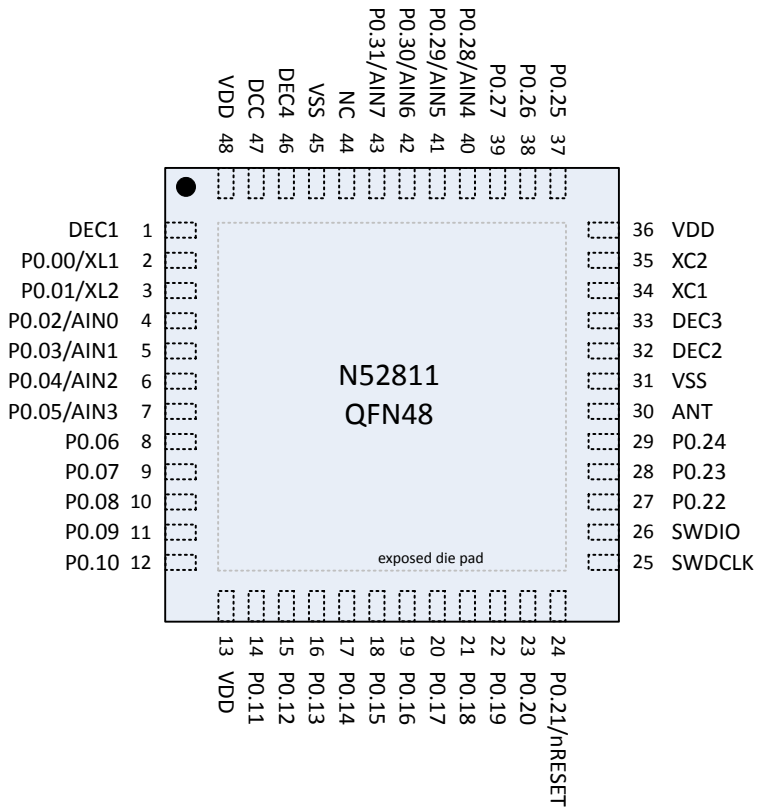



*Figure 151: QFN48 pin assignments, top view*

| Pin | Name | Type | Description |
|---|---|---|---|
| 1 | DEC1 | Power | 0.9 V regulator digital supply decoupling |
| 2 | P0.00 | Digital I/O | General purpose I/O |
| | XL1 | Analog input | Connection for 32.768 kHz crystal (LFXO) |

| Pin | Name | Type | Description |
|---|---|---|---|
| 3 | P0.01 | Digital I/O | General purpose I/O |
| | XL2 | Analog input | Connection for 32.768 kHz crystal (LFXO) |
| 4 | P0.02 | Digital I/O | General purpose I/O |
| | AIN0 | Analog input | COMP input<br>SAADC input |
| 5 | P0.03 | Digital I/O | General purpose I/O |
| | AIN1 | Analog input | COMP input<br>SAADC input |
| 6 | P0.04 | Digital I/O | General purpose I/O |
| | AIN2 | Analog input | COMP input<br>SAADC input |
| 7 | P0.05 | Digital I/O | General purpose I/O |
| | AIN3 | Analog input | COMP input<br>SAADC input |
| 8 | P0.06 | Digital I/O | General purpose I/O |
| 9 | P0.07 | Digital I/O | General purpose I/O |
| 10 | P0.08 | Digital I/O | General purpose I/O |
| 11 | P0.09 | Digital I/O | General purpose I/O |
| 12 | P0.10 | Digital I/O | General purpose I/O |
| 13 | VDD | Power | Power supply |
| 14 | P0.11 | Digital I/O | General purpose I/O |
| 15 | P0.12 | Digital I/O | General purpose I/O |
| 16 | P0.13 | Digital I/O | General purpose I/O |
| 17 | P0.14 | Digital I/O | General purpose I/O |
| 18 | P0.15 | Digital I/O | General purpose I/O |
| 19 | P0.16 | Digital I/O | General purpose I/O |
| 20 | P0.17 | Digital I/O | General purpose I/O |
| 21 | P0.18 | Digital I/O | General purpose I/O |
| 22 | P0.19 | Digital I/O | General purpose I/O |
| 23 | P0.20 | Digital I/O | General purpose I/O |
| 24 | P0.21 | Digital I/O | General purpose I/O |
| | nRESET | | Configurable as pin reset |
| 25 | SWDCLK | Digital input | Serial wire debug clock input for debug and programming |
| 26 | SWDIO | Digital I/O | Serial wire debug I/O for debug and programming |
| 27 | P0.22 | Digital I/O | General purpose I/O |
| 28 | P0.23 | Digital I/O | General purpose I/O |
| 29 | P0.24 | Digital I/O | General purpose I/O |
| 30 | ANT | RF | Single-ended radio antenna connection |
| 31 | VSS | Power | Ground (radio supply) |
| 32 | DEC2 | Power | 1.3 V regulator supply decoupling (radio supply) |
| 33 | DEC3 | Power | Power supply decoupling |
| 34 | XC1 | Analog input | Connection for 32 MHz crystal |
| 35 | XC2 | Analog input | Connection for 32 MHz crystal |
| 36 | VDD | Power | Power supply |

| Pin | Name | Type | Description |
|---|---|---|---|
| 37 | P0.25 | Digital I/O | General purpose I/O |
| 38 | P0.26 | Digital I/O | General purpose I/O |
| 39 | P0.27 | Digital I/O | General purpose I/O |
| 40 | P0.28<br>AIN4 | Digital I/O<br>Analog input | General purpose I/O<br>COMP input<br>SAADC input |
| 41 | P0.29<br>AIN5 | Digital I/O<br>Analog input | General purpose I/O<br>COMP input<br>SAADC input |
| 42 | P0.30<br>AIN6 | Digital I/O<br>Analog input | General purpose I/O<br>COMP input<br>SAADC input |
| 43 | P0.31<br>AIN7 | Digital I/O<br>Analog input | General purpose I/O pin<br>COMP input<br>SAADC input |
| 44 | NC | | No connect<br>Leave unconnected |
| 45 | VSS | Power | Ground |
| 46 | DEC4 | Power | 1.3 V regulator supply decoupling<br>Input from DC/DC regulator<br>Output from 1.3 V LDO |
| 47 | DCC | Power | DC/DC regulator output |
| 48 | VDD | Power | Power supply |
| Die pad | VSS | Power | Ground pad<br>Exposed die pad must be connected to ground (VSS) for proper device operation. |

*Table 120: QFN48 pin assignments*

## 7.1.2 QFN32 pin assignments

The nRF52811 QFN32 pin assignment table and figure describe the pinouts for this variant of the chip.

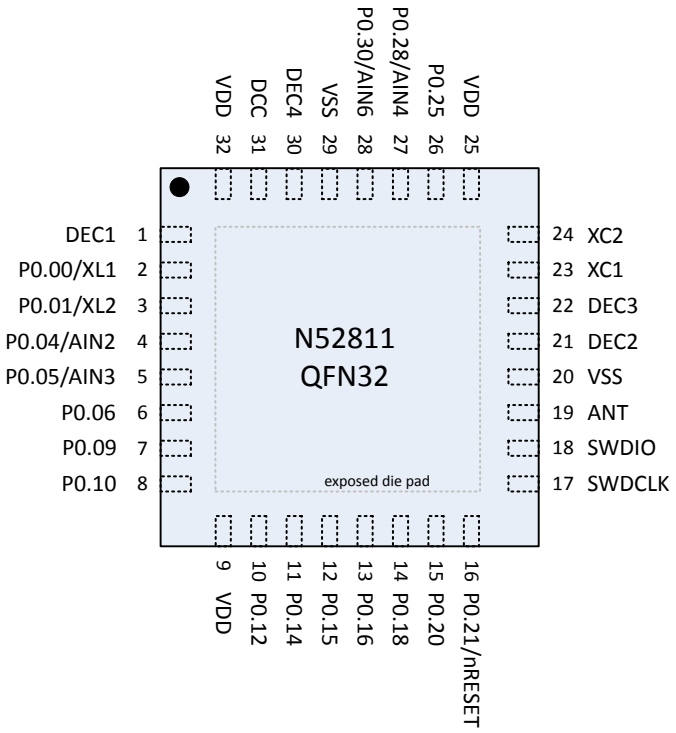



*Figure 152: QFN32 pin assignments, top view*

| Pin | Name | Type | Description |
|---|---|---|---|
| 1 | DEC1 | Power | 0.9 V regulator digital supply decoupling |
| 2 | P0.00 | Digital I/O | General purpose I/O |
| | XL1 | Analog input | Connection for 32.768 kHz crystal (LFXO) |
| 3 | P0.01 | Digital I/O | General purpose I/O |
| | XL2 | Analog input | Connection for 32.768 kHz crystal (LFXO) |
| 4 | P0.04 | Digital I/O | General purpose I/O |
| | AIN2 | Analog input | COMP input<br>SAADC input |
| 5 | P0.05 | Digital I/O | General purpose I/O |
| | AIN3 | Analog input | COMP input<br>SAADC input |
| 6 | P0.06 | Digital I/O | General purpose I/O |
| 7 | P0.09 | Digital I/O | General purpose I/O |
| 8 | P0.10 | Digital I/O | General purpose I/O |
| 9 | VDD | Power | Power supply |
| 10 | P0.12 | Digital I/O | General purpose I/O |
| 11 | P0.14 | Digital I/O | General purpose I/O |
| 12 | P0.15 | Digital I/O | General purpose I/O |

| Pin | Name | Type | Description |
|---|---|---|---|
| 13 | P0.16 | Digital I/O | General purpose I/O |
| 14 | P0.18 | Digital I/O | General purpose I/O<br>Single wire output |
| 15 | P0.20 | Digital I/O | General purpose I/O |
| 16 | P0.21<br>nRESET | Digital I/O | General purpose I/O<br>Configurable as pin reset |
| 17 | SWDCLK | Digital input | Serial wire debug clock input for debug and programming |
| 18 | SWDIO | Digital I/O | Serial wire debug I/O for debug and programming |
| 19 | ANT | RF | Single-ended radio antenna connection |
| 20 | VSS | Power | Ground (radio supply) |
| 21 | DEC2 | Power | 1.3 V regulator supply decoupling (radio supply) |
| 22 | DEC3 | Power | Power supply decoupling |
| 23 | XC1 | Analog input | Connection for 32 MHz crystal |
| 24 | XC2 | Analog input | Connection for 32 MHz crystal |
| 25 | VDD | Power | Power supply |
| 26 | P0.25 | Digital I/O | General purpose I/O |
| 27 | P0.28<br>AIN4 | Digital I/O<br>Analog input | General purpose I/O<br>COMP input<br>SAADC input |
| 28 | P0.30<br>AIN6 | Digital I/O<br>Analog input | General purpose I/O<br>COMP input<br>SAADC input |
| 29 | VSS | Power | Ground |
| 30 | DEC4 | Power | 1.3 V regulator supply decoupling<br>Input from DC/DC regulator<br>Output from 1.3 V LDO |
| 31 | DCC | Power | DC/DC regulator output |
| 32 | VDD | Power | Power supply |
| Die pad | VSS | Power | Ground pad<br>Exposed die pad must be connected to ground (VSS) for proper device operation. |

*Table 121: QFN32 pin assignments*

## 7.1.3 WLCSP ball assignments

The nRF52811 ball assignment table and figure describe the assignments for this variant of the chip.

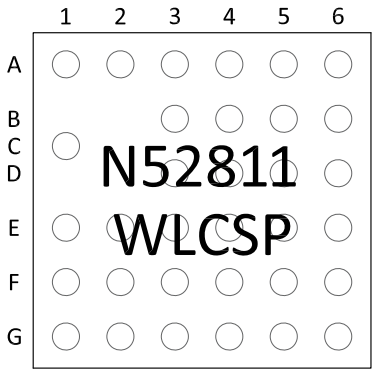



*Figure 153: WLCSP ball assignments, top view*

Balls not mentioned in the ball assignments table below are not connected (NC) and must be soldered to the PCB.

| Pin | Name | Type | Description |
|---|---|---|---|
| A1 | XC1 | Analog input | Connection for 32 MHz crystal |
| A2 | XC2 | Analog input | Connection for 32 MHz crystal |
| A3 | DEC2 | Power | 1.3 V regulator supply decoupling (radio supply) |
| A4 | DEC4 | Power | 1.3 V analog supply.<br>Input from DC/DC converter. Output from 1.3 V LDO. |
| A5 | DCC | Power | DC/DC converter output (3.3 V PWM) |
| A6 | VDD | Power | Power (battery) supply |
| B3 | VSS | Power | Ground |
| B4 | VSS | Power | Ground |
| B5 | P0.00 | Digital I/O | General purpose I/O |
| | XL1 | Analog input | Connection for 32.768 kHz crystal (LFXO) |
| B6 | DEC1 | Power | 0.9 V regulator digital supply decoupling |
| C1 | VSS_PA | Power | Ground |
| D3 | VSS | Power | Ground |
| D4 | VSS | Power | Ground |
| D5 | P0.01 | Digital I/O | General purpose I/O |
| | XL2 | Analog input | Connection for 32.768 kHz crystal (LFXO) |
| D6 | P0.03 | Digital I/O | General purpose I/O |
| | AIN1 | Analog input | SAADC/COMP/LPCOMP input |
| E1 | ANT | RF | Single-ended radio antenna connection |
| E2 | P0.18 | Digital I/O | General purpose I/O |
| E3 | VSS | Power | Ground |
| E4 | VSS | Power | Ground |
| E5 | P0.04 | Digital I/O | General purpose I/O |
| | AIN2 | Analog input | SAADC/COMP/LPCOMP input |
| E6 | P0.05 | Digital I/O | General purpose I/O |
| | AIN3 | Analog input | SAADC/COMP/LPCOMP input |
| F1 | SWDIO | Digital I/O | Serial wire debug I/O for debug and programming |
| F2 | P0.21 | Digital I/O | General purpose I/O |
| | nRESET | | Configurable as pin reset |
| F3 | P0.17 | Digital I/O | General purpose I/O |
| F4 | P0.14 | Digital I/O | General purpose I/O |
| F5 | P0.11 | Digital I/O | General purpose I/O |
| F6 | P0.08 | Digital I/O | General purpose I/O |
| G1 | SWDCLK | Digital input | Serial wire debug clock input for debug and programming |
| G2 | P0.20 | Digital I/O | General purpose I/O |
| G3 | P0.16 | Digital I/O | General purpose I/O |
| G4 | P0.15 | Digital I/O | General purpose I/O |
| G5 | P0.12 | Digital I/O | General purpose I/O |
| G6 | VDD | Power | Power (battery) supply |

*Table 122: WLCSP ball assignments*

### 7.1.4 GPIO pins located near the radio

Radio performance parameters, such as sensitivity, may be affected by high frequency digital I/O with large sink/source current close to the radio power supply and antenna pins.

GPIO recommended usage on page 423 identifies some GPIO pins that have recommended usage guidelines for maximizing radio performance in an application.

| GPIO | QFN48 pin | QFN32 pin | Recommended usage |
|---|---|---|---|
| P0.25 | 37 | 26 | Low drive, low frequency I/O only. |
| P0.26 | 38 | | |
| P0.27 | 39 | | |
| P0.28 | 40 | 27 | |
| P0.29 | 41 | | |

*Table 123: GPIO recommended usage*

## 7.2 Mechanical specifications

The mechanical specifications for the packages show the dimensions in millimeters.

### 7.2.1 QFN48 6 x 6 mm package

Dimensions in millimeters for the nRF52811 QFN48 6 x 6 mm package.

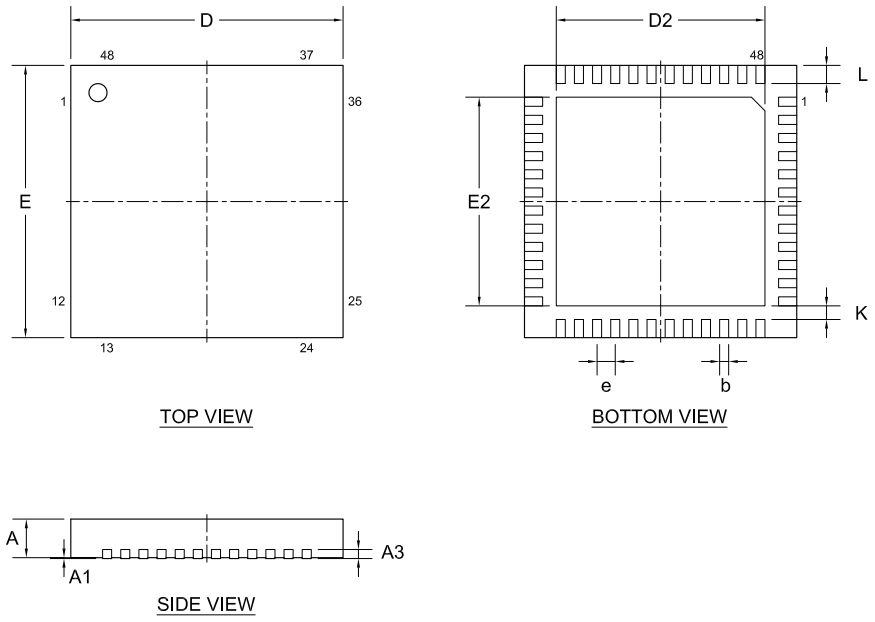



*Figure 154: QFN48 6 x 6 mm package*

| | A | A1 | A3 | b | D, E | D2, E2 | e | K | L |
|---|---|---|---|---|---|---|---|---|---|
| Min. | 0.80 | 0.00 | | 0.15 | | 4.50 | | 0.20 | 0.35 |
| Nom. | 0.85 | 0.04 | 0.20 | 0.20 | 6.00 | 4.60 | 0.40 | | 0.40 |
| Max. | 0.90 | 0.05 | | 0.25 | | 4.70 | | | 0.45 |

*Table 124: QFN48 dimensions in millimeters*

## 7.2.2 QFN32 5 x 5 mm package

Dimensions in millimeters for the nRF52811 QFN32 5 x 5 mm package.

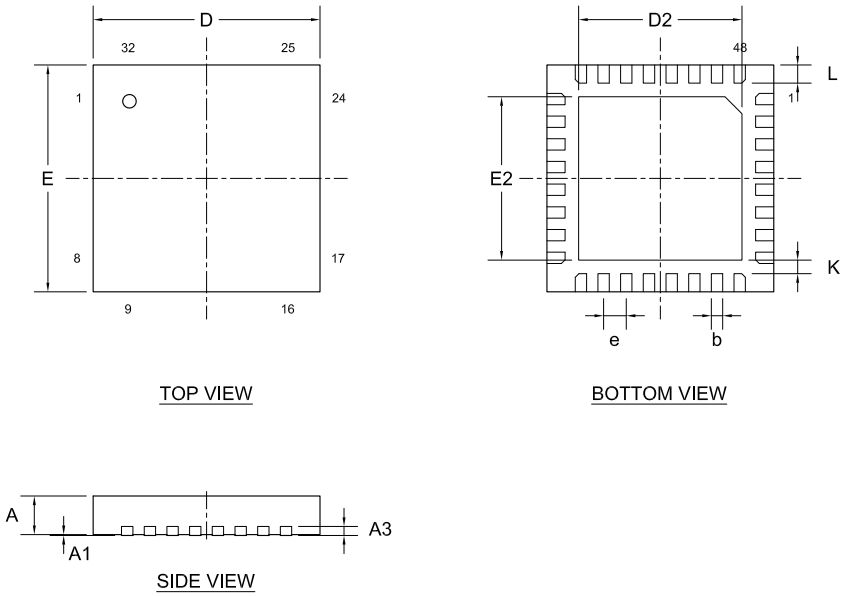



*Figure 155: QFN32 5 x 5 mm package*

| | A | A1 | A3 | b | D, E | D2, E2 | e | K | L |
|---|---|---|---|---|---|---|---|---|---|
| Min. | 0.80 | 0.00 | | 0.20 | | 3.40 | | 0.25 | 0.35 |
| Nom. | 0.85 | 0.04 | 0.20 | 0.25 | 5.00 | 3.50 | 0.50 | | 0.40 |
| Max. | 0.90 | 0.05 | | 0.30 | | 3.60 | | | 0.45 |

*Table 125: QFN32 dimensions in millimeters*

## 7.2.3 WLCSP 2.482 x 2.464 mm package

Dimensions in millimeters for the nRF52811 WLCSP 2.482 x 2.464 mm package.

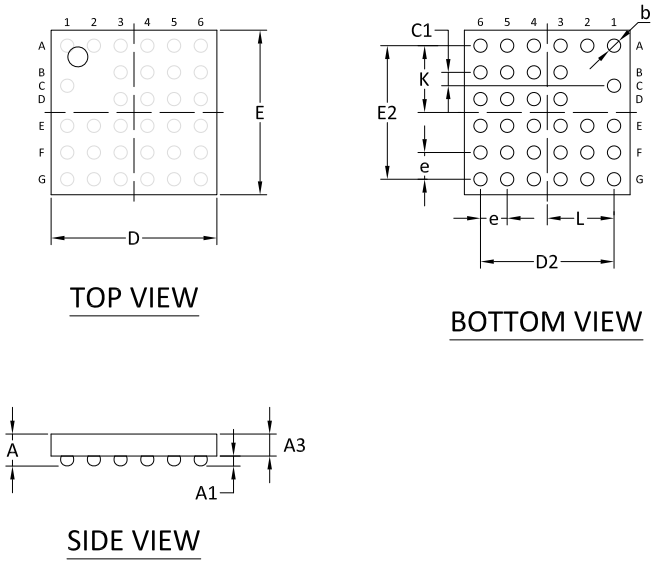



*Figure 156: WLCSP 2.482 x 2.464 mm package*

| | A | A1 | A3 | b | C1 | D | E | D2 | E2 | e | K | L |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Min. | 0.419 | 0.12 | 0.299 | 0.197 | | | | | | | | |
| Nom. | 0.477 | | 0.327 | | 0.2 | 2.482 | 2.464 | 2.0 | 2.0 | 0.4 | 1.0 | 1.0 |
| Max. | 0.535 | 0.18 | 0.355 | 0.257 | | | | | | | | |

*Table 126: WLCSP dimensions in millimeters*

# 7.3 Reference circuitry

To ensure good RF performance when designing PCBs, it is highly recommended to use the PCB layouts and component values provided by Nordic Semiconductor.

Documentation for the different package reference circuits, including Altium Designer files, PCB layout files, and PCB production files can be downloaded from Reference layout nRF52 Series .

## 7.3.1 Schematic QFAA QFN48 with internal LDO regulator setup

In addition to the schematic, the bill of material (BOM) is also provided.

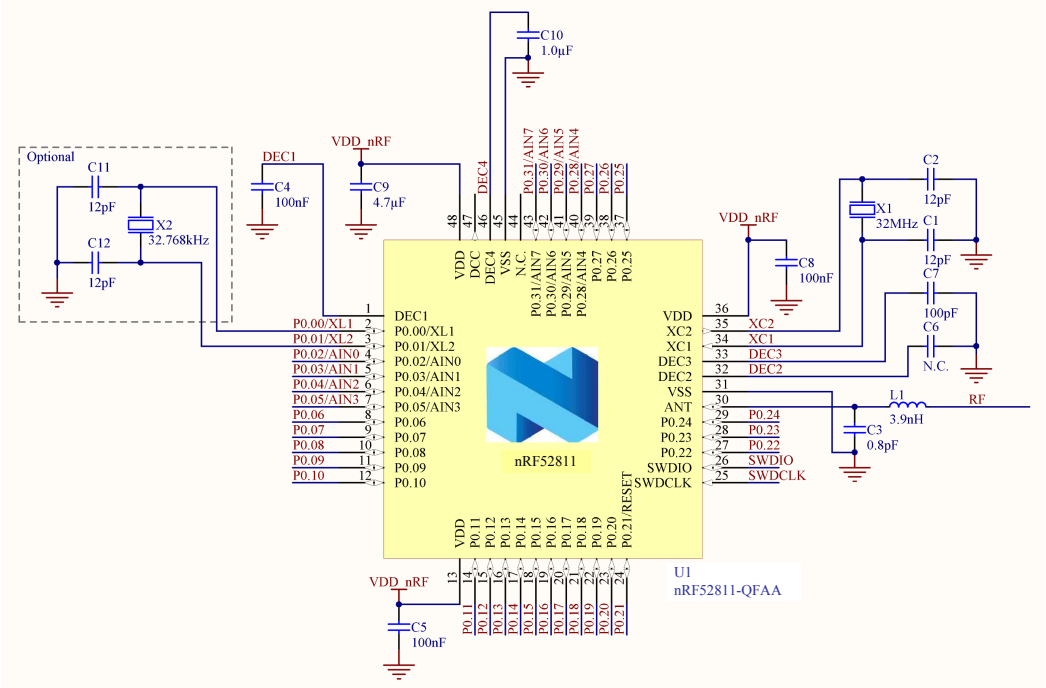



*Figure 157: QFAA QFN48 with internal LDO regulator setup*

**Note:** For PCB reference layouts, see Reference layout nRF52 Series.

| Designator | Value | Description | Footprint |
|---|---|---|---|
| C1, C2, C11, C12 | 12 pF | Capacitor, NP0, ±2% | 0402 |
| C3 | 0.8 pF | Capacitor, NP0, ±5% | 0402 |
| C4, C5, C8 | 100 nF | Capacitor, X7R, ±10% | 0402 |
| C6 | N.C. | Not mounted | 0402 |
| C7 | 100 pF | Capacitor, NP0, ±5% | 0402 |
| C9 | 4.7 µF | Capacitor, X5R, ±10% | 0603 |
| C10 | 1.0 µF | Capacitor, X7R, ±10% | 0603 |
| L1 | 3.9 nH | High frequency chip inductor ±5% | 0402 |
| U1 | nRF52811-QFAA | Multi-protocol Bluetooth® low energy, ANT and 2.4 GHz proprietary system on chip | QFN-48 |
| X1 | 32 MHz | XTAL SMD 2016, 32 MHz, Cl = 8 pF, Total Tol: ±40 ppm | XTAL_2016 |
| X2 | 32.768 kHz | XTAL SMD 3215, 32.768 kHz, Cl = 9 pF, Total Tol: ±20 ppm | XTAL_3215 |

*Table 127: Bill of material for QFAA QFN48 with internal LDO regulator setup*

## 7.3.2 Schematic QFAA QFN48 with DC/DC regulator setup

In addition to the schematic, the bill of material (BOM) is also provided.

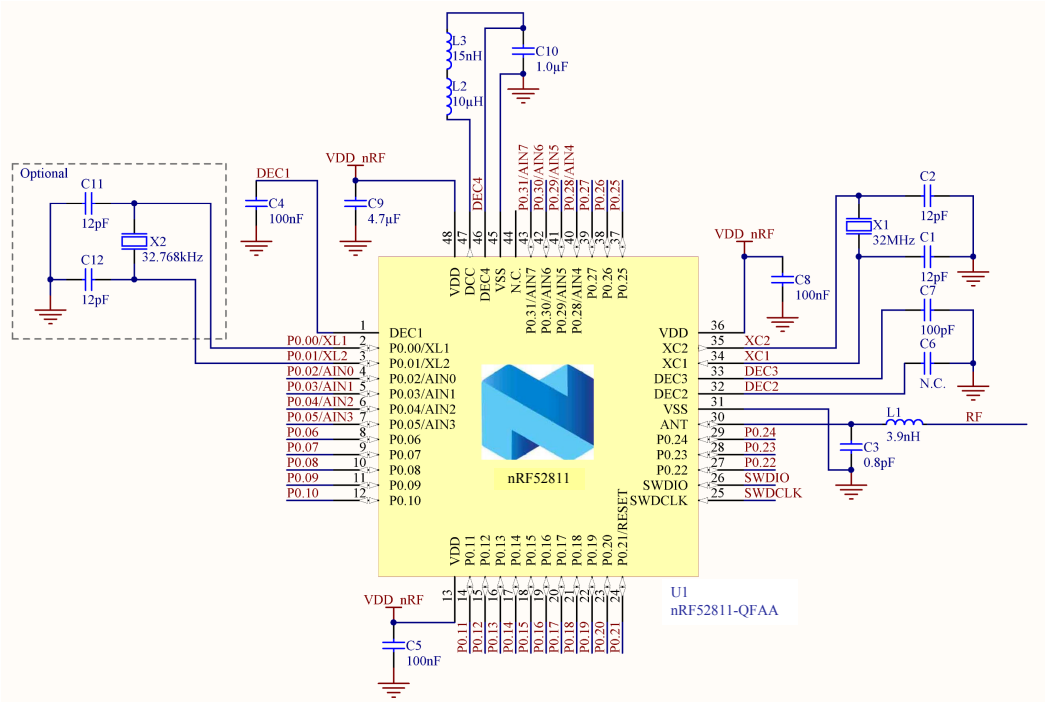



*Figure 158: QFAA QFN48 with DC/DC regulator setup*

**Note:** For PCB reference layouts, see Reference layout nRF52 Series.

| Designator | Value | Description | Footprint |
|---|---|---|---|
| C1, C2, C11, C12 | 12 pF | Capacitor, NP0, ±2% | 0402 |
| C3 | 0.8 pF | Capacitor, NP0, ±5% | 0402 |
| C4, C5, C8 | 100 nF | Capacitor, X7R, ±10% | 0402 |
| C6 | N.C. | Not mounted | 0402 |
| C7 | 100 pF | Capacitor, NP0, ±5% | 0402 |
| C9 | 4.7 µF | Capacitor, X5R, ±10% | 0603 |
| C10 | 1.0 µF | Capacitor, X7R, ±10% | 0603 |
| L1 | 3.9 nH | High frequency chip inductor ±5% | 0402 |
| L2 | 10 µH | Chip inductor, IDC,min = 50 mA, ±20% | 0603 |
| L3 | 15 nH | High frequency chip inductor ±10% | 0402 |
| U1 | nRF52811-QFAA | Multi-protocol Bluetooth® low energy, ANT and 2.4 GHz proprietary system on chip | QFN-48 |
| X1 | 32 MHz | XTAL SMD 2016, 32 MHz, Cl = 8 pF, Total Tol: ±40 ppm | XTAL_2016 |
| X2 | 32.768 kHz | XTAL SMD 3215, 32.768 kHz, Cl = 9 pF, Total Tol: ±20 ppm | XTAL_3215 |

*Table 128: Bill of material for QFAA QFN48 with DC/DC regulator setup*

## 7.3.3 Schematic QCAA QFN32 with internal LDO regulator setup

In addition to the schematic, the bill of material (BOM) is also provided.

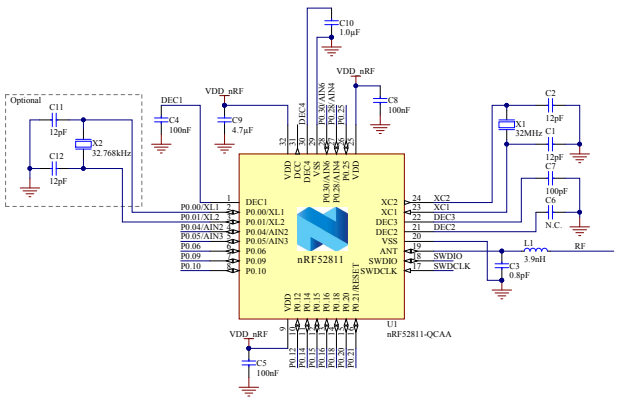



*Figure 159: QCAA QFN32 with internal LDO regulator setup*

**Note:** For PCB reference layouts, see Reference layout nRF52 Series.

| Designator | Value | Description | Footprint |
|---|---|---|---|
| C1, C2, C11, C12 | 12 pF | Capacitor, NP0, ±2% | 0402 |
| C3 | 0.8 pF | Capacitor, NP0, ±5% | 0402 |
| C4, C5, C8 | 100 nF | Capacitor, X7R, ±10% | 0402 |
| C6 | N.C. | Not mounted | 0402 |
| C7 | 100 pF | Capacitor, NP0, ±5% | 0402 |
| C9 | 4.7 µF | Capacitor, X5R, ±10% | 0603 |
| C10 | 1.0 µF | Capacitor, X7R, ±10% | 0603 |
| L1 | 3.9 nH | High frequency chip inductor ±5% | 0402 |
| U1 | nRF52811-QCAA | Multi-protocol Bluetooth® low energy, ANT and 2.4 GHz proprietary system on chip | QFN-32 |
| X1 | 32 MHz | XTAL SMD 2016, 32 MHz, Cl = 8 pF, Total Tol: ±40 ppm | XTAL_2016 |
| X2 | 32.768 kHz | XTAL SMD 3215, 32.768 kHz, Cl = 9 pF, Total Tol: ±20 ppm | XTAL_3215 |

*Table 129: Bill of material for QCAA QFN32 with internal LDO regulator setup*

## 7.3.4 Schematic QCAA QFN32 with DC/DC regulator setup

In addition to the schematic, the bill of material (BOM) is also provided.

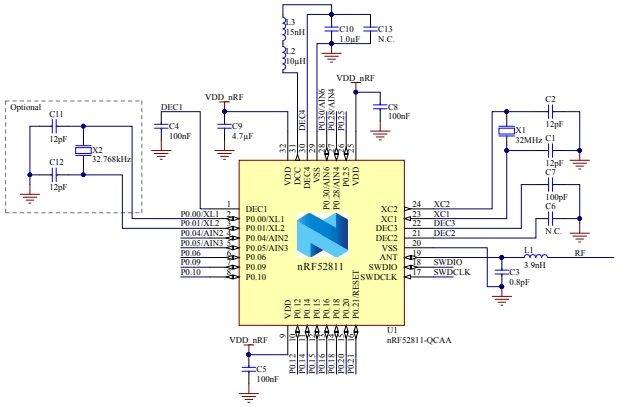



*Figure 160: QCAA QFN32 with DC/DC regulator setup*

**Note:** For PCB reference layouts, see Reference layout nRF52 Series.

| Designator | Value | Description | Footprint |
|---|---|---|---|
| C1, C2, C11, C12 | 12 pF | Capacitor, NP0, ±2% | 0402 |
| C3 | 0.8 pF | Capacitor, NP0, ±5% | 0402 |
| C4, C5, C8 | 100 nF | Capacitor, X7R, ±10% | 0402 |
| C6 | N.C. | Not mounted | 0402 |
| C7 | 100 pF | Capacitor, NP0, ±5% | 0402 |
| C9 | 4.7 µF | Capacitor, X5R, ±10% | 0603 |
| C10 | 1.0 µF | Capacitor, X7R, ±10% | 0603 |
| L1 | 3.9 nH | High frequency chip inductor ±5% | 0402 |
| L2 | 10 µH | Chip inductor, IDC,min = 50 mA, ±20% | 0603 |
| L3 | 15 nH | High frequency chip inductor ±10% | 0402 |
| U1 | nRF52811-QCAA | Multi-protocol Bluetooth® low energy, ANT and 2.4 GHz proprietary system on chip | QFN-32 |
| X1 | 32 MHz | XTAL SMD 2016, 32 MHz, Cl = 8 pF, Total Tol: ±40 ppm | XTAL_2016 |
| X2 | 32.768 kHz | XTAL SMD 3215, 32.768 kHz, Cl = 9 pF, Total Tol: ±20 ppm | XTAL_3215 |

*Table 130: Bill of material for QCAA QFN32 with DC/DC regulator setup*

## 7.3.5 Schematic CAAA WLCSP with internal LDO regulator setup

In addition to the schematic, the bill of material (BOM) is also provided.

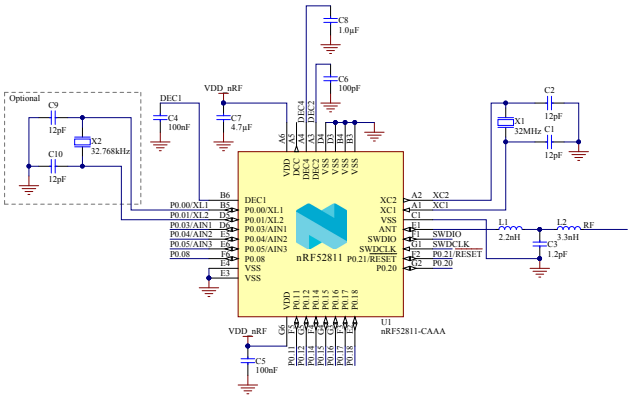



*Figure 161: CAAA WLCSP with internal LDO regulator setup*

**Note:** For PCB reference layouts, see Reference layout nRF52 Series.

| Designator | Value | Description | Footprint |
|---|---|---|---|
| C1, C2, C9, C10 | 12 pF | Capacitor, NP0, ±2 % | 0201 |
| C3 | 1.2 pF | Capacitor, NP0, ±5 % | 0201 |
| C4, C5 | 100 nF | Capacitor, X5R, ±10 % | 0201 |
| C6 | 100 pF | Capacitor, NP0, ±2 % | 0201 |
| C7 | 4.7 µF | Capacitor, X5R, ±10 % | 0603 |
| C8 | 1.0 µF | Capacitor, X5R, ±5 % | 0402 |
| L1 | 2.2 nH | High frequency chip inductor ±5 % | 0201 |
| L2 | 3.3 nH | High frequency chip inductor ±5 % | 0201 |
| U1 | nRF52811-CAAA | Multi-protocol Bluetooth® low energy, ANT and 2.4 GHz proprietary system on chip | WLCSP-33 |
| X1 | 32 MHz | XTAL SMD 2016, 32 MHz, Cl = 8 pF, Total Tol: ±40 ppm | XTAL_2016 |
| X2 | 32.768 kHz | XTAL SMD 2012, 32.768 kHz, Cl = 9 pF, Total Tol: ±50 ppm | XTAL_2012 |

*Table 131: Bill of material for CAAA WLCSP with internal LDO regulator setup*

## 7.3.6 Schematic CAAA WLCSP with DC/DC regulator setup

In addition to the schematic, the bill of material (BOM) is also provided.

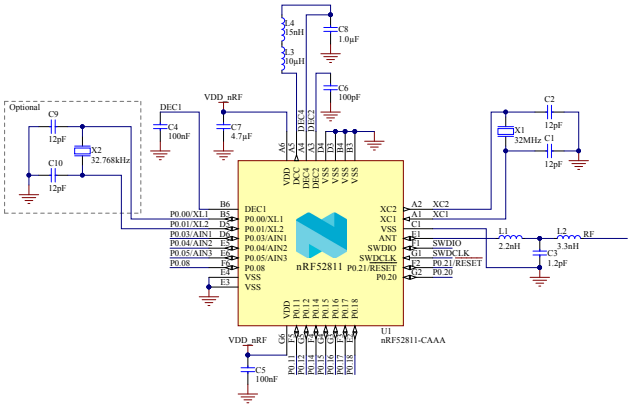



*Figure 162: CAAA WLCSP with DC/DC regulator setup*

**Note:** For PCB reference layouts, see Reference layout nRF52 Series.

| Designator | Value | Description | Footprint |
|---|---|---|---|
| C1, C2, C9, C10 | 12 pF | Capacitor, NP0, ±2 % | 0201 |
| C3 | 1.2 pF | Capacitor, NP0, ±5 % | 0201 |
| C4, C5 | 100 nF | Capacitor, X5R, ±10 % | 0201 |
| C6 | 100 pF | Capacitor, NP0, ±2 % | 0201 |
| C7 | 4.7 µF | Capacitor, X5R, ±10 % | 0603 |
| C8 | 1.0 µF | Capacitor, X5R, ±5 % | 0402 |
| L1 | 2.2 nH | High frequency chip inductor ±5 % | 0201 |
| L2 | 3.3 nH | High frequency chip inductor ±5 % | 0201 |
| L3 | 10 µH | Chip inductor, IDC,min = 50 mA, ±20 % | 0603 |
| L4 | 15 nH | High frequency chip inductor ±10 % | 0402 |
| U1 | nRF52811-CAAA | Multi-protocol Bluetooth® low energy, ANT and 2.4 GHz proprietary system on chip | WLCSP-33 |
| X1 | 32 MHz | XTAL SMD 2016, 32 MHz, Cl = 8 pF, Total Tol: ±40 ppm | XTAL_2016 |
| X2 | 32.768 kHz | XTAL SMD 2012, 32.768 kHz, Cl = 9 pF, Total Tol: ±50 ppm | XTAL_2012 |

*Table 132: Bill of material for CAAA WLCSP with DC/DC regulator setup*

## 7.3.7 Schematic CAAA WLCSP with two layers

In addition to the schematic, the bill of material (BOM) is also provided.

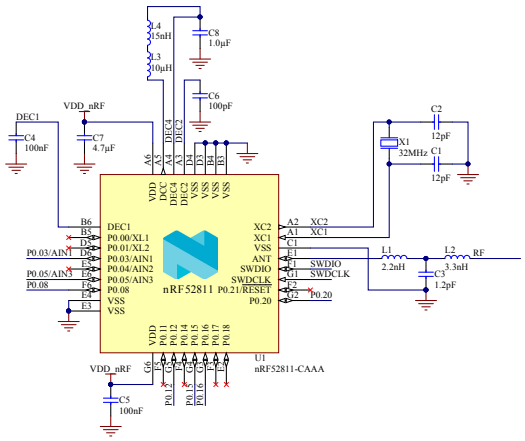



*Figure 163: CAAA WLCSP 2-layer setup*

**Note:** For PCB reference layouts, see Reference layout nRF52 Series.

| Designator | Value | Description | Footprint |
|---|---|---|---|
| C1, C2 | 12 pF | Capacitor, NP0, ±2 % | 0201 |
| C3 | 1.2 pF | Capacitor, NP0, ±5 % | 0201 |
| C4, C5 | 100 nF | Capacitor, X5R, ±10 % | 0201 |
| C6 | 100 pF | Capacitor, NP0, ±2 % | 0201 |
| C7 | 4.7 µF | Capacitor, X5R, ±10 % | 0603 |
| C8 | 1.0 µF | Capacitor, X5R, ±5 % | 0402 |
| L1 | 2.2 nH | High frequency chip inductor ±5 % | 0201 |
| L2 | 3.3 nH | High frequency chip inductor ±5 % | 0201 |
| L3 | 10 µH | Chip inductor, IDC,min = 50 mA, ±20 % | 0603 |
| L4 | 15 nH | High frequency chip inductor ±10 % | 0402 |
| U1 | nRF52811-CAAA | Multi-protocol Bluetooth® low energy, ANT and 2.4 GHz proprietary system on chip | WLCSP-33 |
| X1 | 32 MHz | XTAL SMD 2016, 32 MHz, Cl = 8 pF, Total Tol: ±40 ppm | XTAL_2016 |

*Table 133: Bill of material for CAAA WLCSP 2-layer setup*

## 7.3.8 PCB guidelines

A well designed PCB is necessary to achieve good RF performance. Poor layout can lead to loss in performance or functionality.

A qualified RF layout for the IC and its surrounding components, including matching networks, can be downloaded from Reference layout nRF52 Series .

To ensure optimal performance it is essential that you follow the schematics and layout references closely. Especially in the case of the antenna matching circuitry (components between device pin ANT and the antenna), any changes to the layout can change the behavior, resulting in degradation of RF performance or a need to change component values. All reference circuits are designed for use with a 50 Ω single-ended antenna.

A PCB with a minimum of two layers, including a ground plane, is recommended for optimal performance. On PCBs with more than two layers, put a keep-out area on the inner layers directly below the antenna matching circuitry (components between device pin ANT and the antenna) to reduce the stray capacitances that influence RF performance.

A matching network is needed between the RF pin ANT and the antenna, to match the antenna impedance (normally 50 Ω) to the optimum RF load impedance for the chip. For optimum performance, the impedance for the matching network should be set as described in the recommended package reference circuitry in Reference circuitry on page 425 above.

The DC supply voltage should be decoupled as close as possible to the VDD pins with high performance RF capacitors. See the schematics for recommended decoupling capacitor values. The supply voltage for the chip should be filtered and routed separately from the supply voltages of any digital circuitry.

Long power supply lines on the PCB should be avoided. All device grounds, VDD connections, and VDD bypass capacitors must be connected as close as possible to the IC. For a PCB with a topside RF ground plane, the VSS pins should be connected directly to the ground plane. For a PCB with a bottom ground plane, the best technique is to have via holes as close as possible to the VSS pads. A minimum of one via hole should be used for each VSS pin.

Fast switching digital signals should not be routed close to the crystal or the power supply lines. Capacitive loading of fast switching digital output lines should be minimized in order to avoid radio interference.

## 7.3.9 PCB layout example

The PCB layout shown as the example is a reference layout for the QFN48 package with internal LDO setup.

**Important:** Pay attention to how the capacitor C3 is grounded. It is not directly connected to the ground plane, but grounded via VSS pin 31. This is done to create additional filtering of harmonic components.

For all available reference layouts, see Reference layout nRF52 Series.

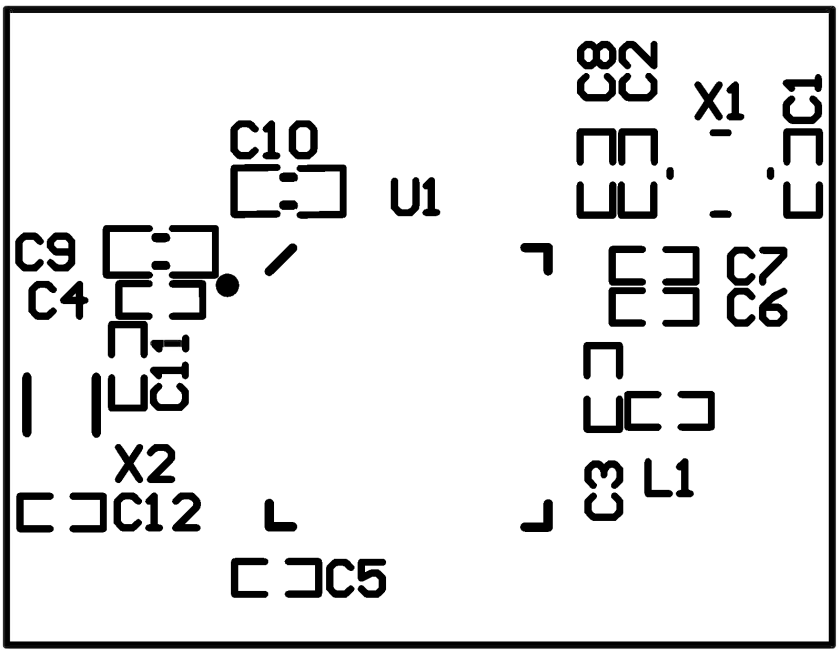



*Figure 164: Top silk layer*

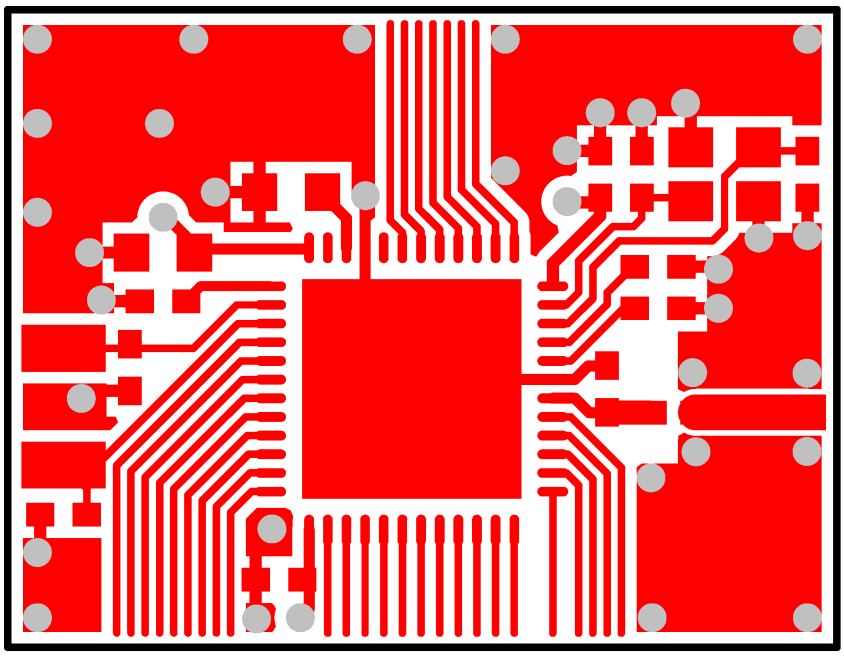


*Figure 165: Top layer*

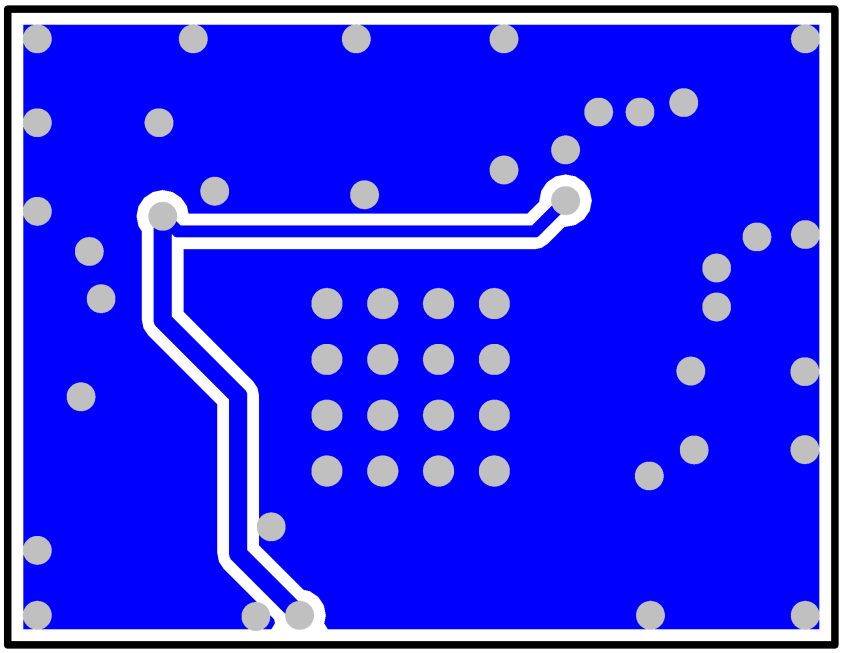


*Figure 166: Bottom layer*

**Important:** No components in bottom layer.

# 8 Recommended operating conditions

The operating conditions are the physical parameters that the chip can operate within.

| Symbol | Parameter | Notes | Min. | Nom. | Max. | Units |
|---|---|---|---|---|---|---|
| VDD | Supply voltage, independent of DCDC enable | | 1.7 | 3.0 | 3.6 | V |
| $t_{R\_VDD}$ | Supply rise time (0 V to 1.7 V) | | | | 60 | ms |
| TA | Operating temperature | | -40 | 25 | 85 | °C |

*Table 134: Recommended operating conditions*

**Important:** The on-chip power-on reset circuitry may not function properly for rise times longer than the specified maximum.

## 8.1 WLCSP light sensitivity

WLCSP package variants are sensitive to visible and near infrared light, which means that a final product design must shield the chip properly.

For the CAAA package variant, the marking side is covered with a light absorbing film, while the side edges of the chip and the ball side must be protected by coating or other means.

# 9 Absolute maximum ratings

Maximum ratings are the extreme limits to which the chip can be exposed for a limited amount of time without permanently damaging it. Exposure to absolute maximum ratings for prolonged periods of time may affect the reliability of the device.

| | Note | Min. | Max. | Unit |
|---|---|---|---|---|
| **Supply voltages** | | | | |
| VDD | | -0.3 | +3.9 | V |
| VSS | | | 0 | V |
| **I/O pin voltage** | | | | |
| $V_{I/O}$, VDD ≤ 3.6 V | | -0.3 | VDD + 0.3 | V |
| $V_{I/O}$, VDD > 3.6 V | | -0.3 | 3.9 | V |
| **Radio** | | | | |
| RF input level | | | 10 | dBm |
| **Environmental QFN package** | | | | |
| Storage temperature | | -40 | +125 | °C |
| MSL | Moisture Sensitivity Level | | 2 | |
| ESD HBM | Human Body Model | | 3 | kV |
| ESD HBM Class | Human Body Model Class | | 2 | |
| ESD CDM | Charged Device Model | | 1 | kV |
| **Environmental WLCSP 2.482 x 2.464 mm package** | | | | |
| Storage temperature | | -40 | +125 | °C |
| MSL | Moisture Sensitivity Level | | 1 | |
| ESD HBM | Human Body Model | | 2 | kV |
| ESD HBM Class | Human Body Model Class | | 2 | |
| ESD CDM | Charged Device Model | | 1 | kV |
| **Flash memory** | | | | |
| Endurance | | 10 000 | | Write/erase cycles |
| Retention | | 10 years at 40°C | | |

*Table 135: Absolute maximum ratings*



# 10 Ordering information

This chapter contains information on IC marking, ordering codes, and container sizes.

## 10.1 IC marking

The IC package is marked like described below.

| | | | | | |
|---|---|---|---|---|---|
| N | 5 | 2 | 8 | 1 | 1 |
| <P | P> | <V | V> | <H> | <P> |
| <Y | Y> | <W | W> | <L | L> |

*Figure 167: Package marking*

## 10.2 Box labels

Here are the box labels used for the IC.

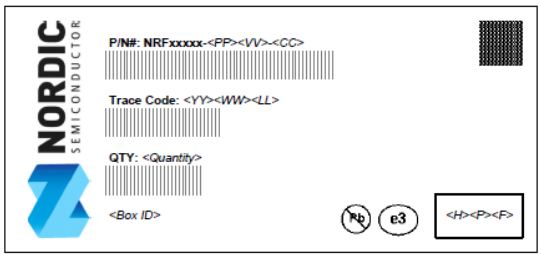



*Figure 168: Inner box label*

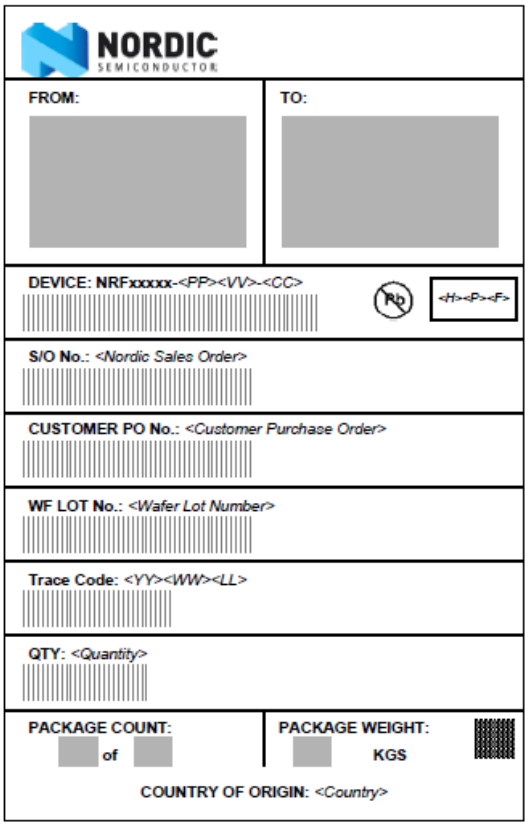



*Figure 169: Outer box label*

## 10.3 Order code

Here are the nRF52811 order codes and definitions.

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| n | R | F | 5 | 2 | 8 | 1 | 1 | - | <P | P> | <V | V> | - | <C | C> |

*Figure 170: Order code*

| Abbrevitation | Definition and implemented codes |
|---|---|
| N52/nRF52 | nRF52 Series product |
| 811 | Part code |
| <PP> | Package variant code |
| <VV> | Function variant code |
| <H><P><F> | Build code<br>H - Hardware version code<br>P - Production configuration code (production site, etc.)<br>F - Firmware version code (only visible on shipping container label) |
| <YY><WW><LL> | Tracking code<br>YY - Year code<br>WW - Assembly week number<br>LL - Wafer lot code |
| <CC> | Container code |

*Table 136: Abbreviations*

## 10.4 Code ranges and values

Defined here are the nRF52811 code ranges and values.

| <PP> | Package | Size (mm) | Pin/Ball count | Pitch (mm) |
|---|---|---|---|---|
| QF | QFN | 6 x 6 | 48 | 0.4 |
| QC | QFN | 5 x 5 | 32 | 0.5 |
| CA | WLCSP | 2.482 x 2.464 | 33 | 0.4 |

*Table 137: Package variant codes*

| <VV> | Flash (kB) | RAM (kB) |
|---|---|---|
| AA | 192 | 24 |

*Table 138: Function variant codes*

| <H> | Description |
|---|---|
| [A . . Z] | Hardware version/revision identifier (incremental) |

*Table 139: Hardware version codes*

| <P> | Description |
|---|---|
| [0 . . 9] | Production device identifier (incremental) |
| [A . . Z] | Engineering device identifier (incremental) |

*Table 140: Production configuration codes*

| <F> | Description |
|---|---|
| [A . . N, P . . Z] | Version of preprogrammed firmware |
| [0] | Delivered without preprogrammed firmware |

*Table 141: Production version codes*

| <YY> | Description |
|---|---|
| [15 . . 99] | Production year: 2015 to 2099 |

*Table 142: Year codes*

| <WW> | Description |
|---|---|
| [1 . . 52] | Week of production |

*Table 143: Week codes*

| <LL> | Description |
|---|---|
| [AA . . ZZ] | Wafer production lot identifier |

*Table 144: Lot codes*

| <CC> | Description |
|---|---|
| R7 | 7" Reel |
| R | 13" Reel |
| T | Tray |

*Table 145: Container codes*

## 10.5 Product options

Defined here are the nRF52811 product options.

| Order code | MOQ (minimum ordering quantity) | Comment |
|---|---|---|
| nRF52811-QFAA-R7 | 1000 | Availability to be announced. |
| nRF52811-QFAA-R | 3000 | |
| nRF52811-QFAA-T | 490 | |
| nRF52811-QCAA-R7 | 1500 | |
| nRF52811-QCAA-R | 4000 | |
| nRF52811-QCAA-T | 490 | |
| nRF52811-CAAA-R7 | 1500 | |
| nRF52811-CAAA-R | 7000 | |

*Table 146: nRF IC order codes*

| Order code | Description |
|---|---|
| nRF52840-DK | nRF52840 development kit with tools to support nRF52811 development. |

*Table 147: Development tools order code*

# 11 Legal notices

By using this documentation you agree to our terms and conditions of use. Nordic Semiconductor ASA may change these terms and conditions at any time without notice.

## 11.1 Liability disclaimer

Nordic Semiconductor ASA reserves the right to make changes without further notice to the product to improve reliability, function or design. Nordic Semiconductor ASA does not assume any liability arising out of the application or use of any product or circuits described herein.

Information in this document is believed to be accurate and reliable. Nordic Semiconductor ASA does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. If there are any discrepancies, ambiguities or conflicts in Nordic Semiconductor's documentation, the Product Specification prevails.

Nordic Semiconductor ASA reserves the right to make corrections, enhancements, and other changes to this document without notice.

## 11.2 Life support applications

Nordic Semiconductor ASA products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury.

Nordic Semiconductor ASA customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Nordic Semiconductor ASA for any damages resulting from such improper use or sale.

## 11.3 RoHS and REACH statement

Nordic Semiconductor products meet the requirements of Directive 2002/95/EC of the European Parliament and of the Council on the Restriction of Hazardous Substances (RoHS) and the requirements of the REACH regulation (EC 1907/2006) on Registration, Evaluation, Authorization and Restriction of Chemicals.

The SVHC (Substances of Very High Concern) candidate list is continually being updated. Complete hazardous substance reports, material composition reports and latest version of Nordic Semiconductor's REACH statement can be found on our website http://www.nordicsemi.com.

## 11.4 Trademarks

All trademarks, service marks, trade names, product names and logos appearing in this documentation are the property of their respective owners.

## 11.5 Copyright notice

© 2018 Nordic Semiconductor ASA. All rights are reserved. Reproduction in whole or in part is prohibited without the prior written permission of the copyright holder.

COMPANY WITH
QUALITY SYSTEM
CERTIFIED BY DNV GL
= ISO 9001 =

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